Retrospective clinical data harmonisation reporting using R and Quarto

Jeremy Selva

@JauntyJJS

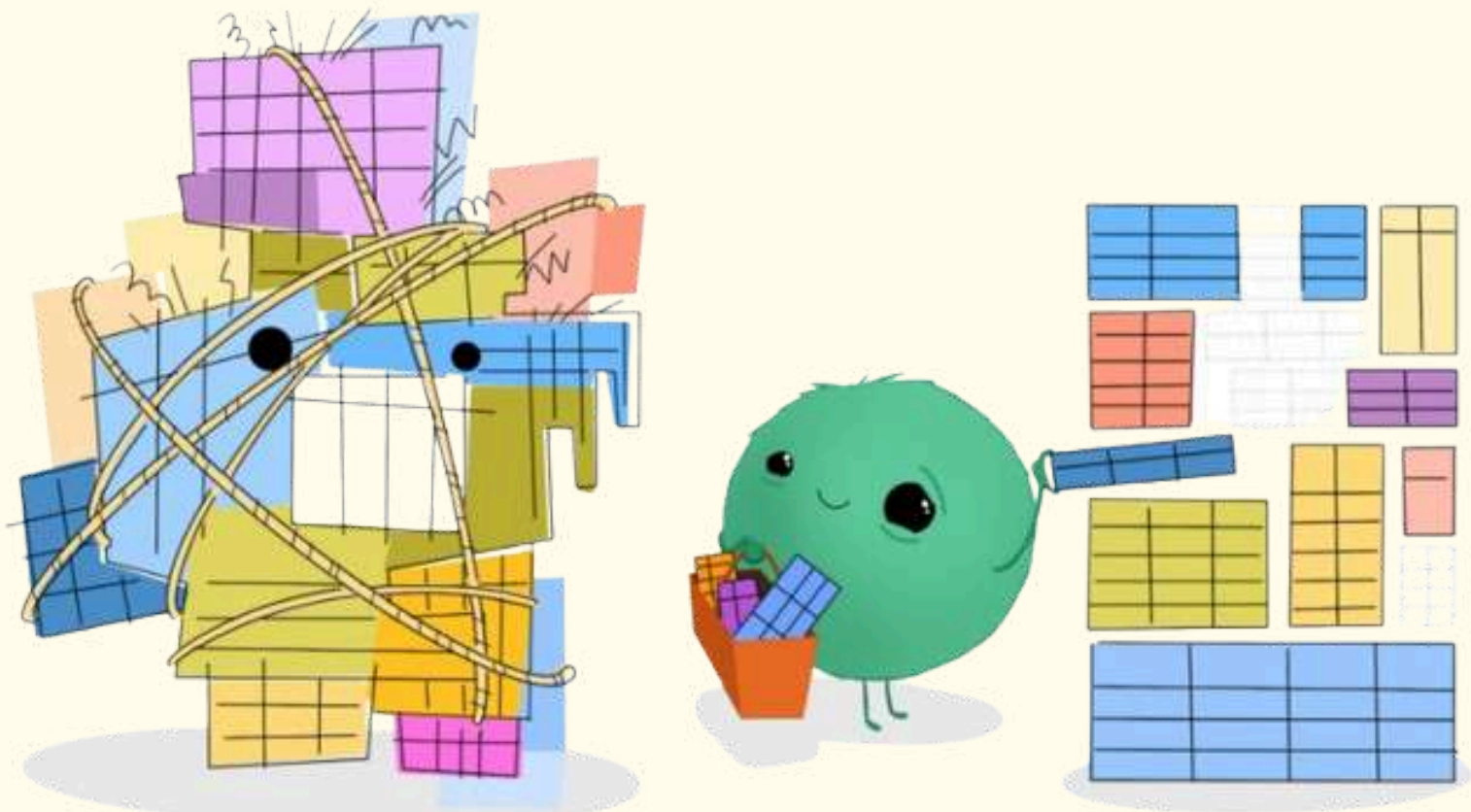
https://jeremy-selva.netlify.app

For useR! 2025

9 August 2025

whoami

Research Officer from National Heart Centre Singapore who collects, cleans and harmonises clinical data.



Taming the Data Beast from "Cleaning Medical Data with R" workshop by Shannon Pileggi, Crystal Lewis and Peter Higgins presented at R/Medicine 2023.

Illustrated by Allison Horst.

About Data Harmonisation

Data harmonisation is part of data wrangling process where

- Similar variables from different datasets are identified.
- Grouped based on a generalised concept they represent.
- Transformed into unified harmonised variables for analysis.

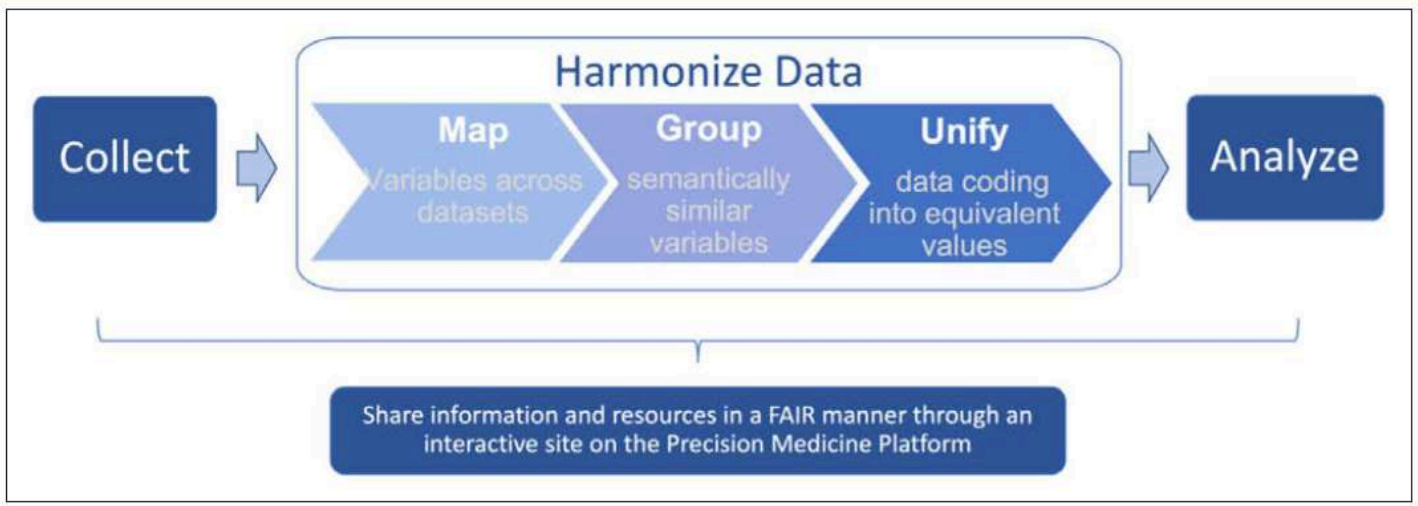
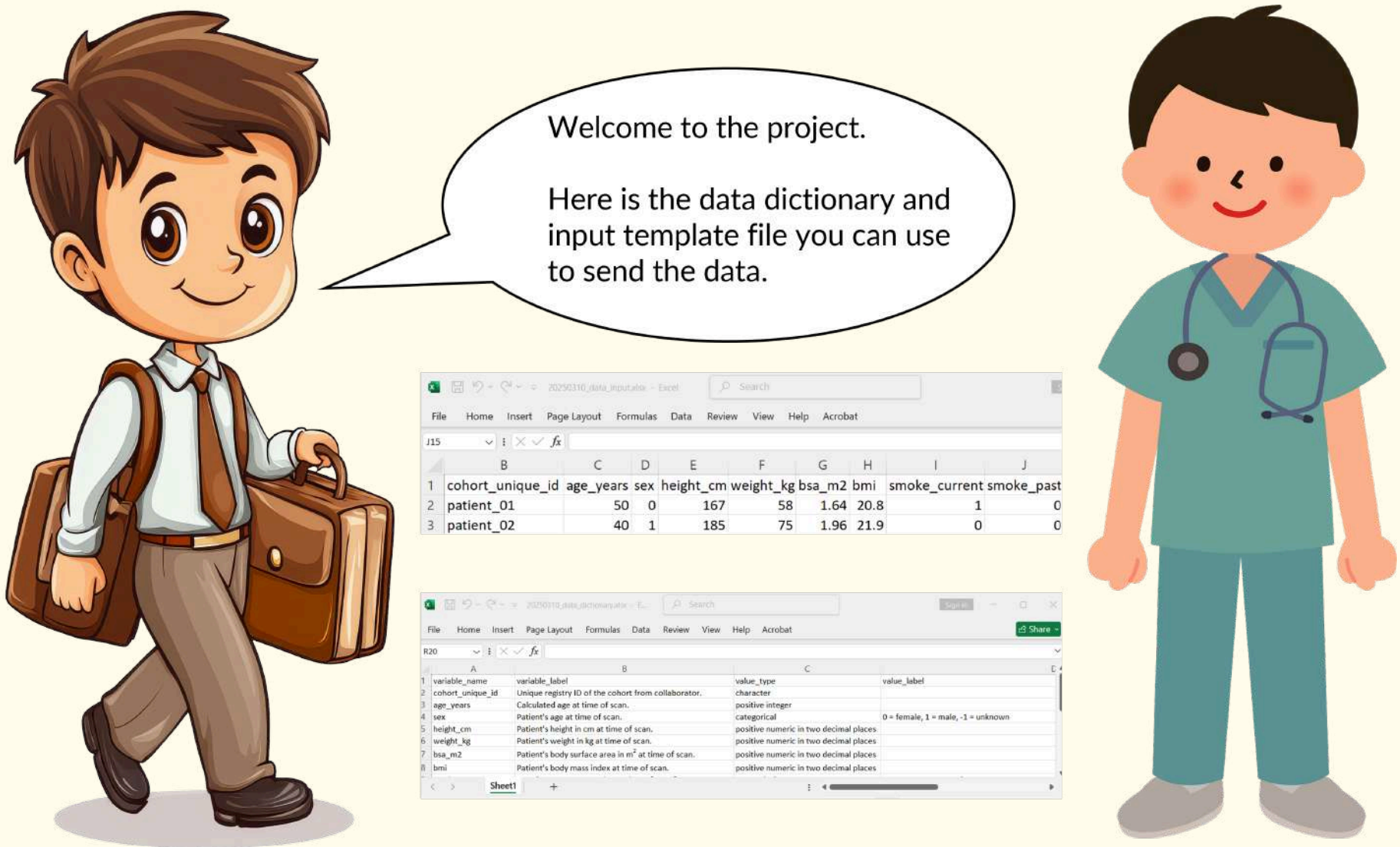


Figure 1. The data harmonization process.
Study data variables collected from different sources need to be mapped to one another (step 1), classified into the generalized concepts they represent (step 2), and transformed into unified harmonized variables (step 3) for analysis.

Image from Mallya et al. Circ Cardiovasc Qual Outcomes. 2023 Nov; 16(11):e009938 doi: 10.1161/CIRCOUTCOMES.123.009938.

How it started

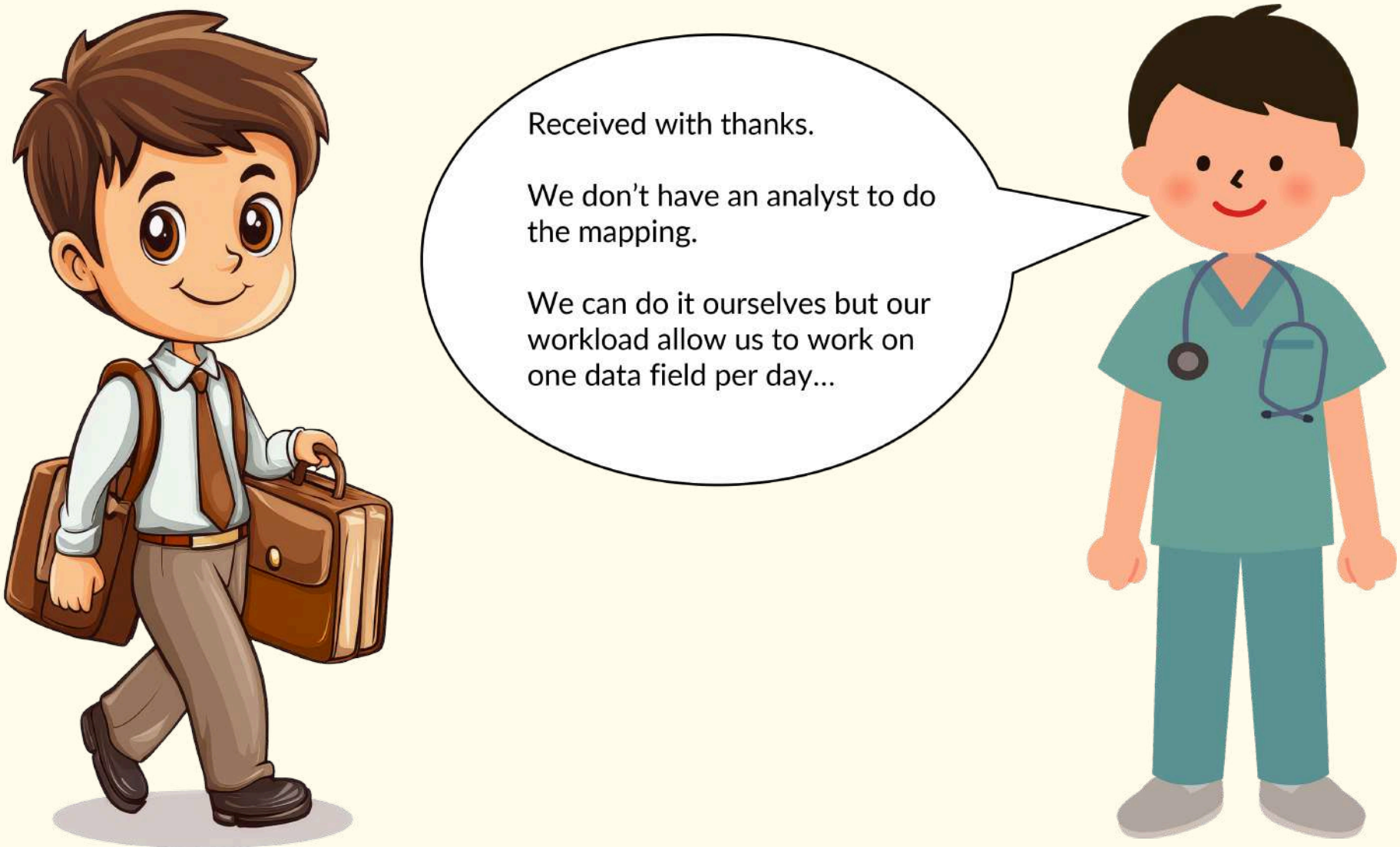


	B	C	D	E	F	G	H	I	J
1	cohort_unique_id	age_years	sex	height_cm	weight_kg	bsa_m2	bmi	smoke_current	smoke_past
2	patient_01	50	0	167	58	1.64	20.8	1	0
3	patient_02	40	1	185	75	1.96	21.9	0	0

	A	B	C	D
1	variable_name	variable_label	value_type	value_label
2	cohort_unique_id	Unique registry ID of the cohort from collaborator.	character	
3	age_years	Calculated age at time of scan.	positive integer	
4	sex	Patient's age at time of scan.	categorical	0 = female, 1 = male, -1 = unknown
5	height_cm	Patient's height in cm at time of scan.	positive numeric in two decimal places	
6	weight_kg	Patient's weight in kg at time of scan.	positive numeric in two decimal places	
7	bsa_m2	Patient's body surface area in m^2 at time of scan.	positive numeric in two decimal places	
8	bmi	Patient's body mass index at time of scan.	positive numeric in two decimal places	

Cheerful Businessman designed by Iftikhar Alam from Vecteezy and Medical Doctor Man from Creazilla.

How it started

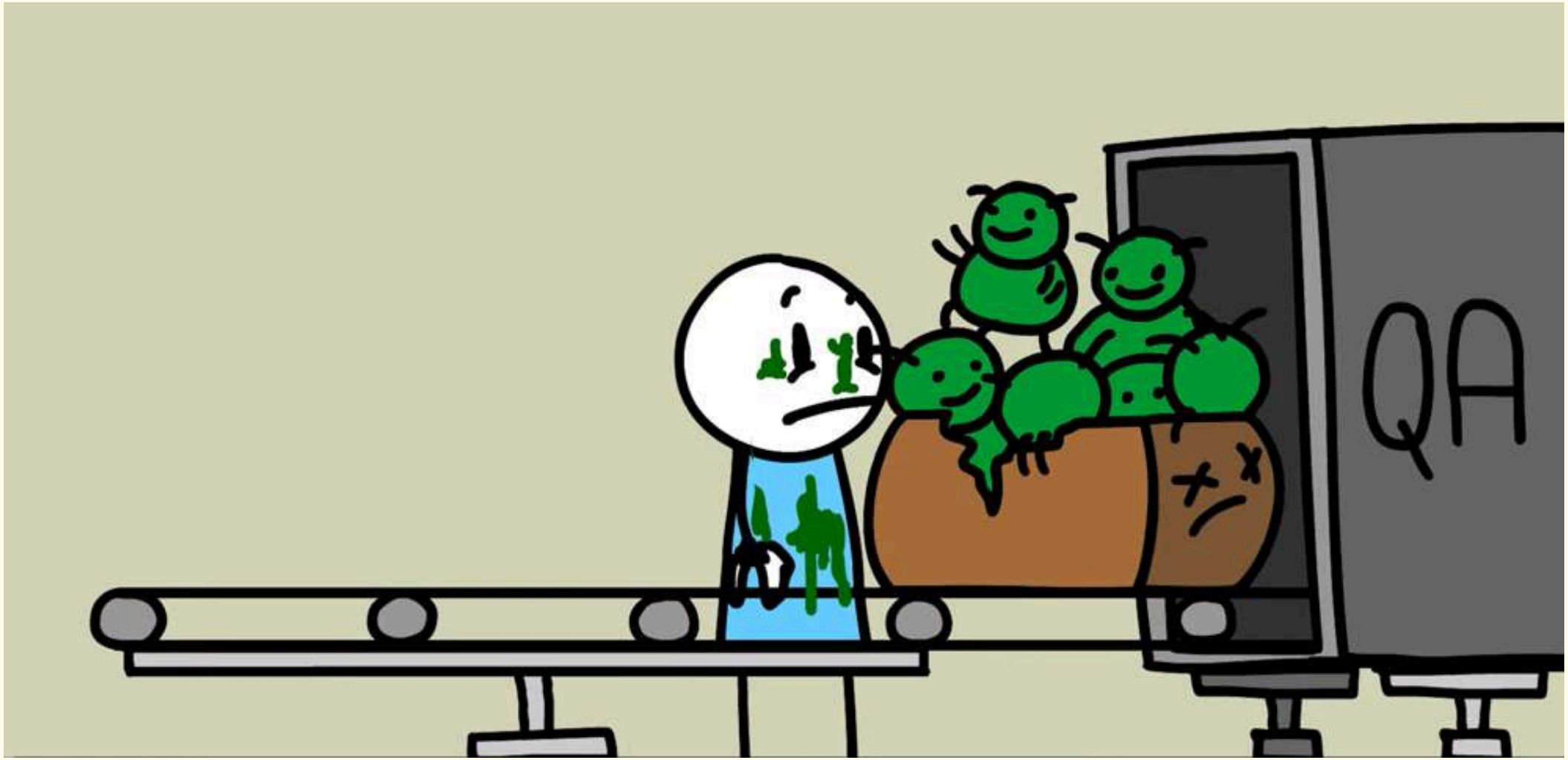


Cheerful Businessman designed by Iftikhar Alam from Vecteezy and Medical Doctor Man from Creazilla.

How it started

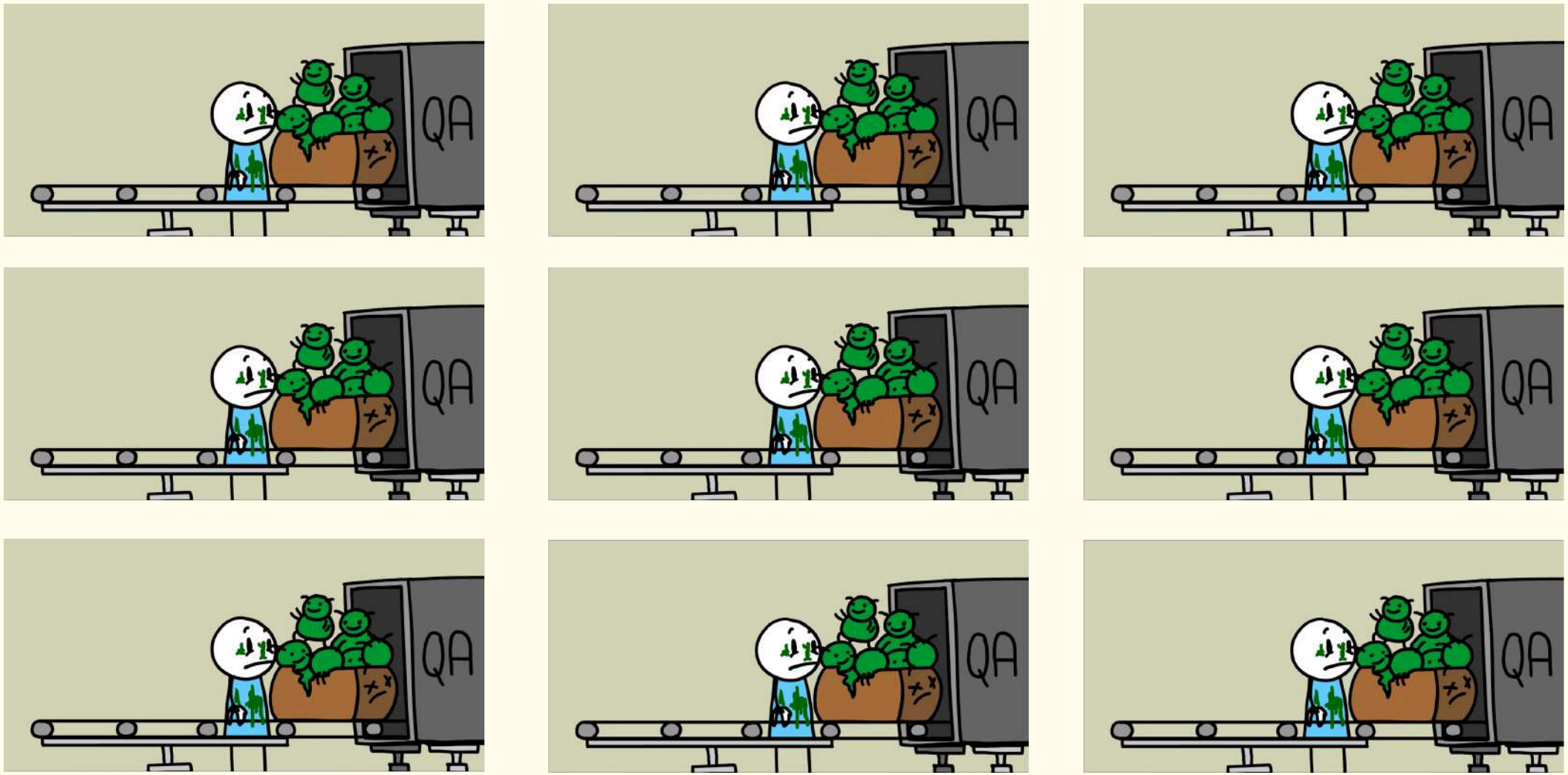
Cheerful Businessman designed by Iftikhar Alam from Vecteezy and Medical Doctor Man from Creazilla.

How it started



snapshot from Ready for QA | MonkeyUser 2SP Animation Video from MonkeyUser.com.

How it started



snapshot from Ready for QA | MonkeyUser 2SP Animation Video from MonkeyUser.com.

How it started

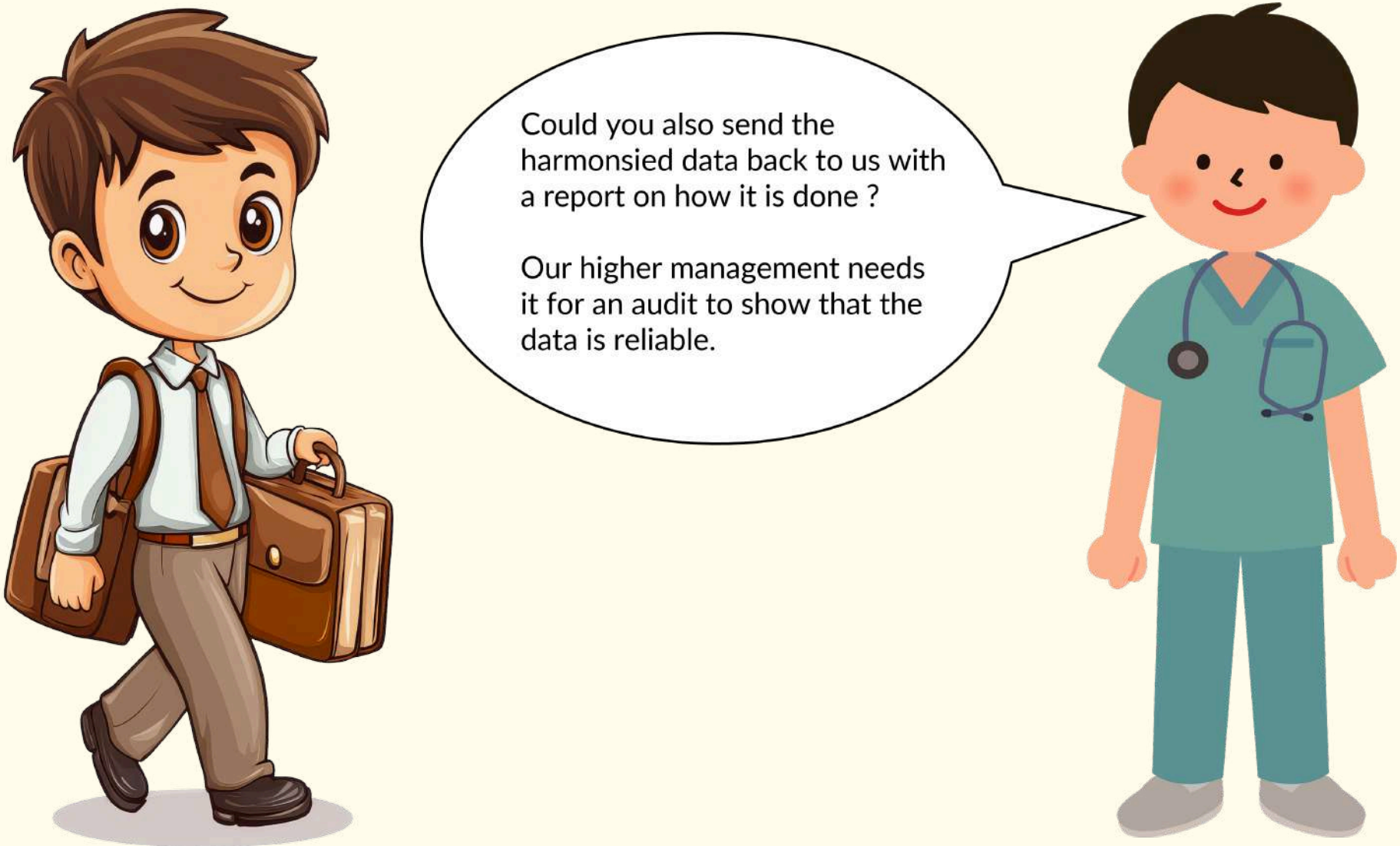
Turn my sorrow into opportunities.

Tackling Formatted Tabular Data from Excel

10th July 2024

Jeremy Selva
@JauntyJJS
https://jeremy-selva.netlify.app

How it started



Cheerful Businessman designed by Iftikhar Alam from Vecteezy and Medical Doctor Man from Creazilla.

Additional Motivation

Some data fields just cannot be planned in advanced.

Cohort 1 Race/Ethnicity
Chinese
Indian
Malay
Eurasian
Others

Cohort 2 Race/Ethnicity
White
Black
Asian
Mixed
Others

Cohort 3 Race/Ethnicity	
Race	Ethnicity
White	Hispanic/Latino
Black	Not Hispanic/Latino
Asian	
Native American	
Pacific Islanders	
Others	

Cohort 4 Race/Ethnicity in text	
Latino	Brazilian
White	British
Asian	Chinese
Middle Eastern	Egyptian
Asian	Korean
Asian	Filipino
White	German
Asian	Japanese
Asian	Korean
Asian	Pakistani
Latino	Peruvian
Middle Eastern	Saudi Arabian
African	Ugandan
etc ...	

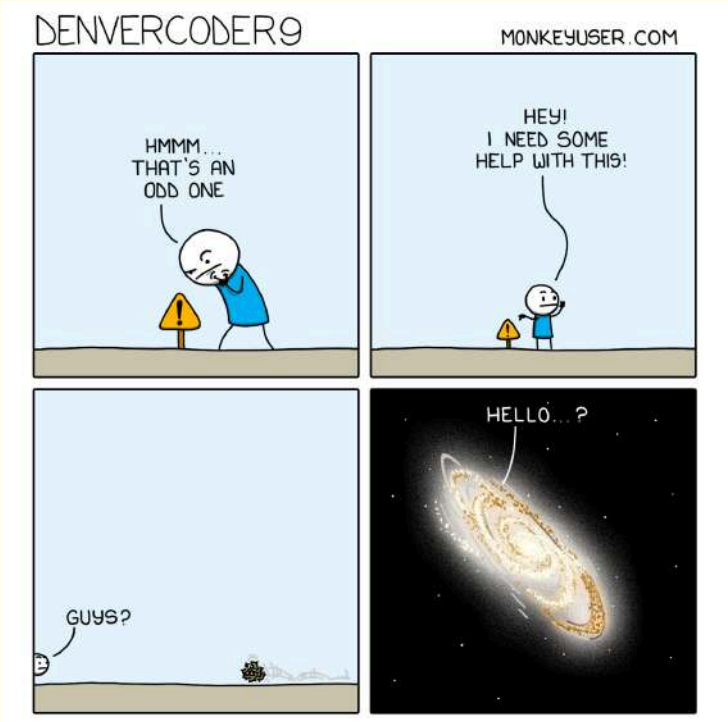
New Cohort Race/Ethnicity
White
African
Southeast Asian
East Asian
South Asian
Other Asians
Middle Eastern
Torres Straits Islanders
Aboriginal
Others

Issues

While there are R packages to facilitate data harmonisation,

- `retroharmonize` for survey data.
- `Rmonize` for epidemiological data.
- `psHarmonize` for health and education data.

There are limited resources on how to make a data harmonisation report.



Denvercoder9 from MonkeyUser.com

Harmonisation Project Template

A template to offer a systematic way to report data harmonisation processes.

[Link: https://jauntyjjs.github.io/harmonisation/]

harmonisation 1.0.0.0 Reference

Search for

Data Harmonisation Project Template

Table of Content

Links

Browse source code

Report a bug

License

Full license

MIT + file LICENSE

Citation

Citing harmonisation

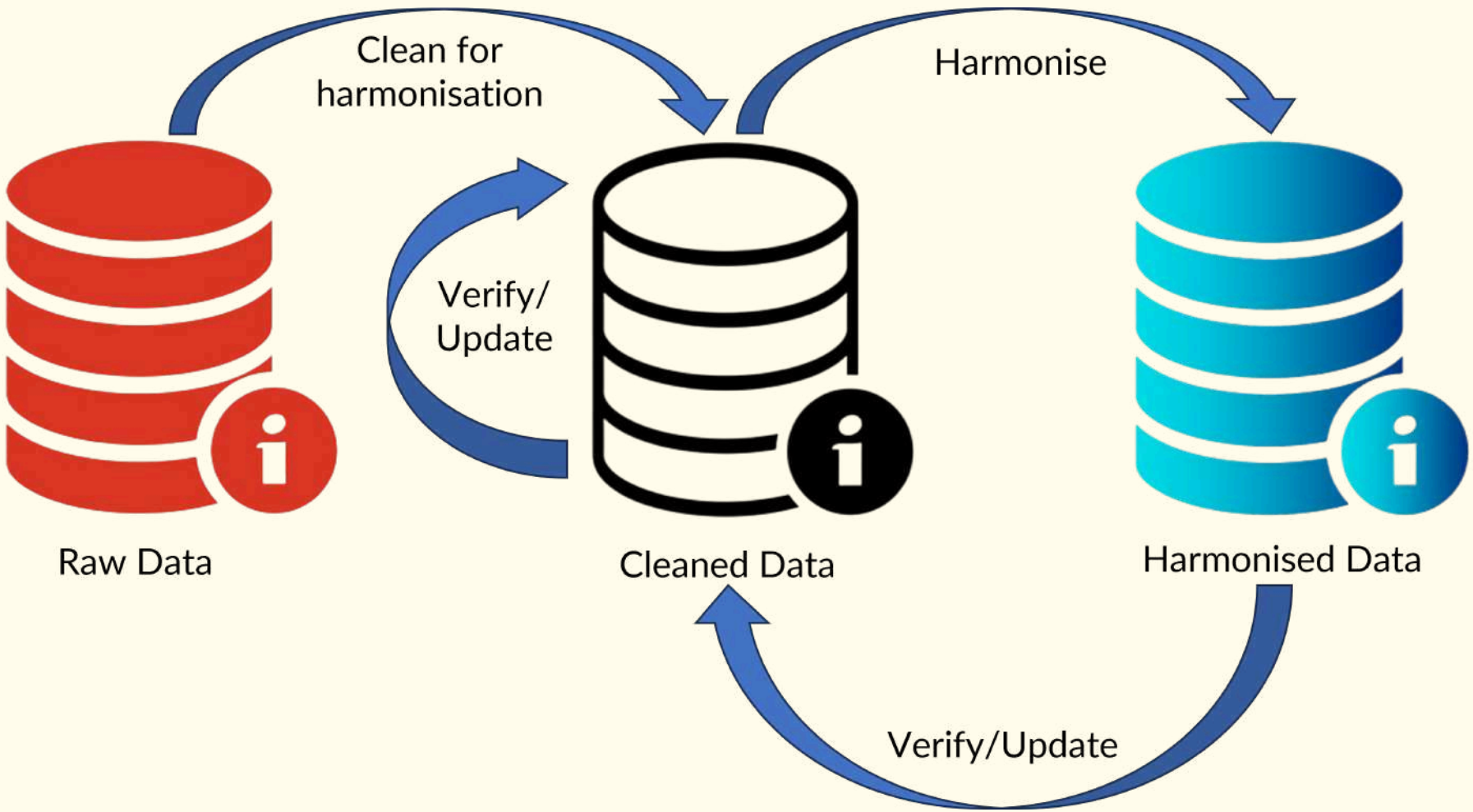
Developers

Jeremy Selva

Author, maintainer

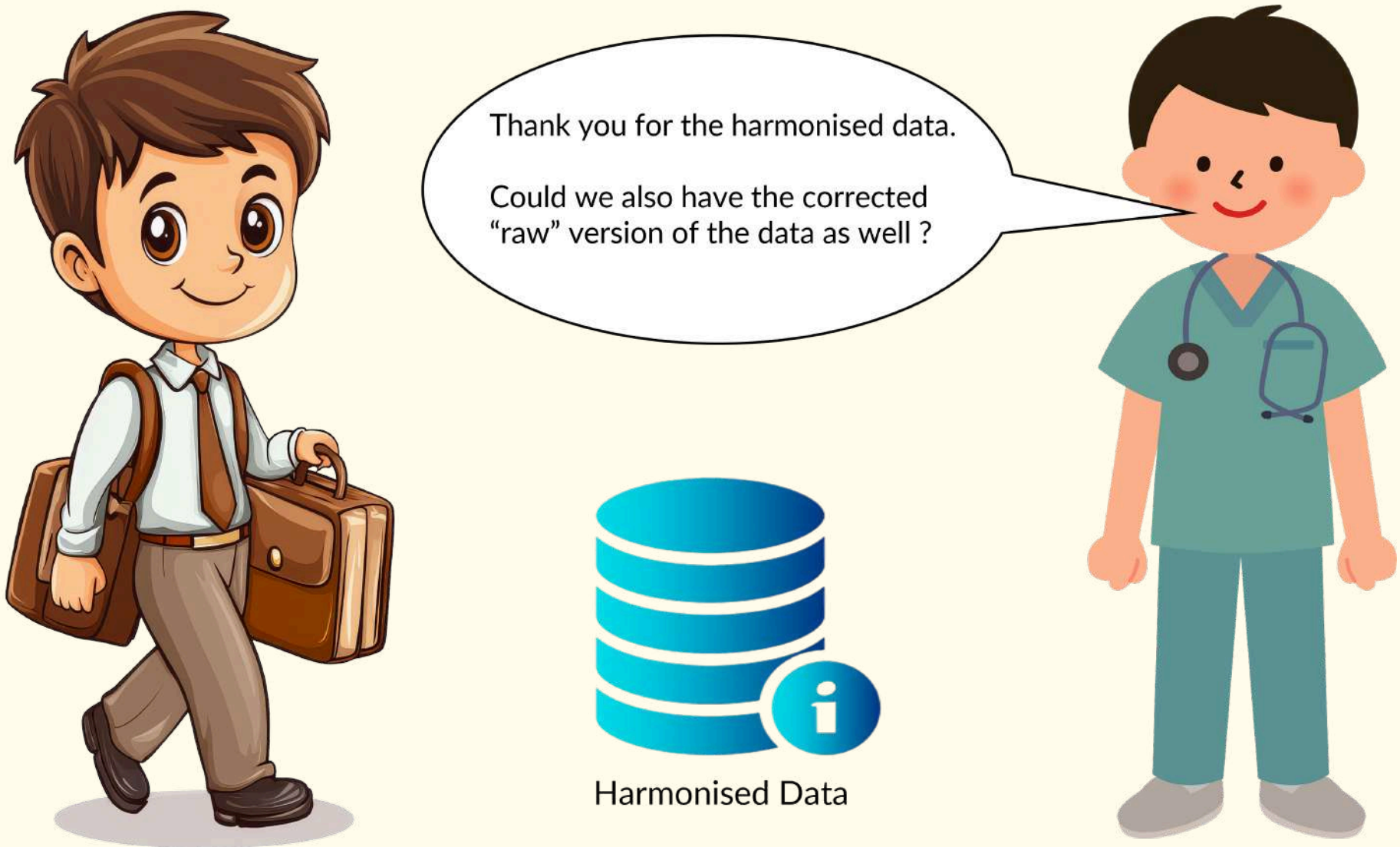
Workflow with collaborators

Collaborator can send the raw data once and you keep updating the cleaned data for harmonsation.



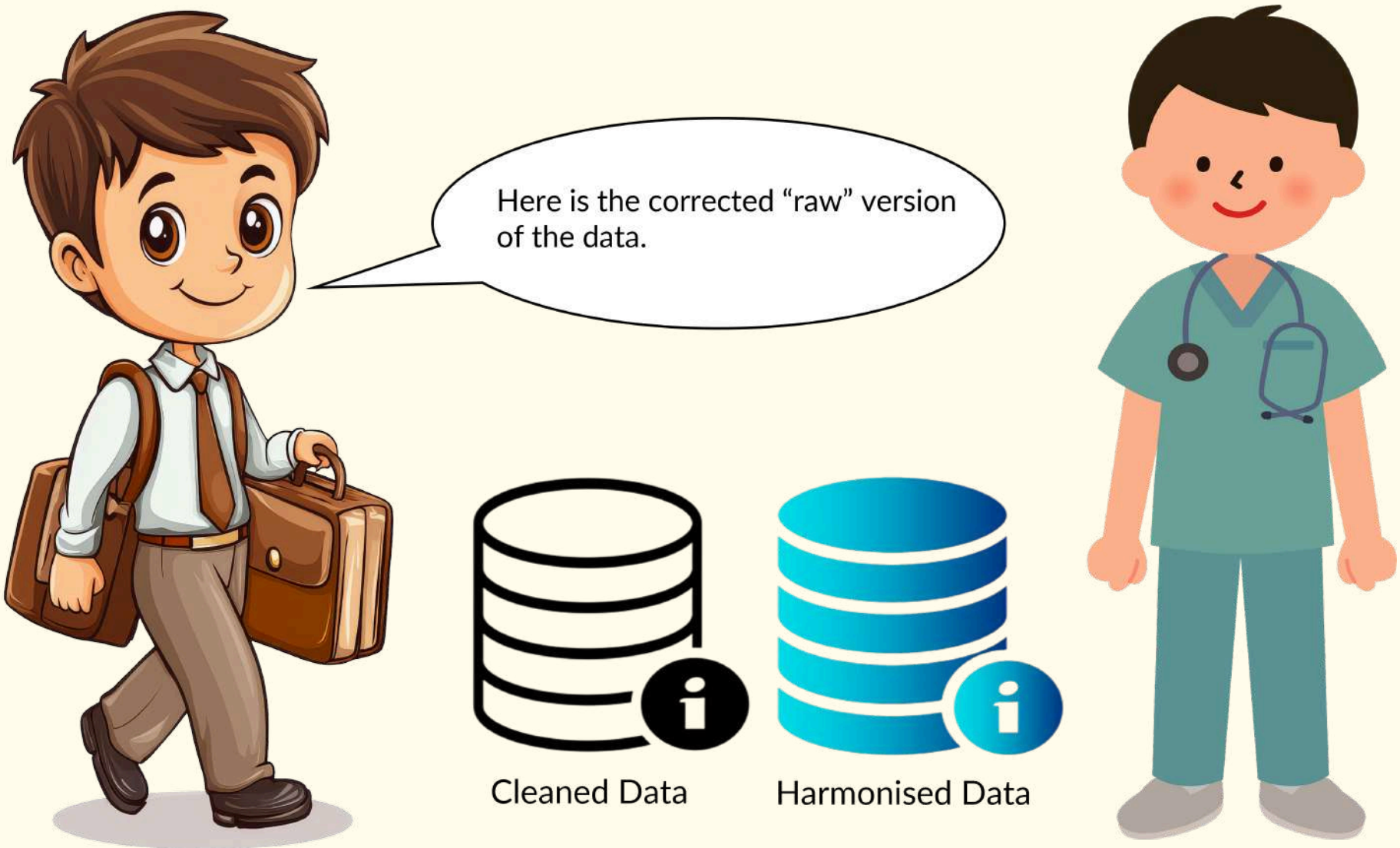
Dataset vectors by Vectora Artworks from Vecteezy.

Workflow with collaborators



Cheerful Businessman designed by Iftikhar Alam from Vecteezy and Medical Doctor Man from Creazilla.

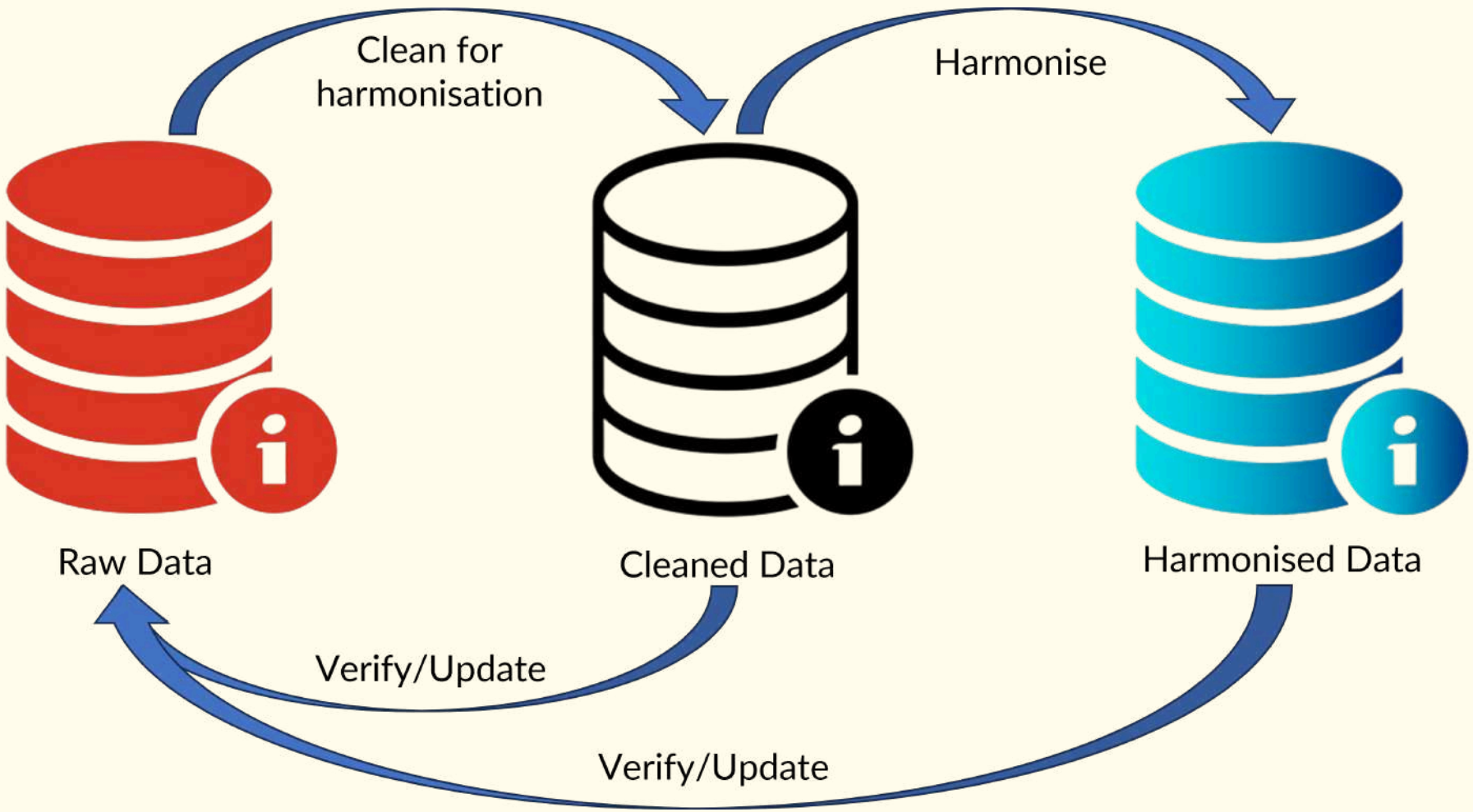
Workflow with collaborators



Cheerful Businessman designed by Iftikhar Alam from Vecteezy and Medical Doctor Man from Creazilla.

Workflow with collaborators

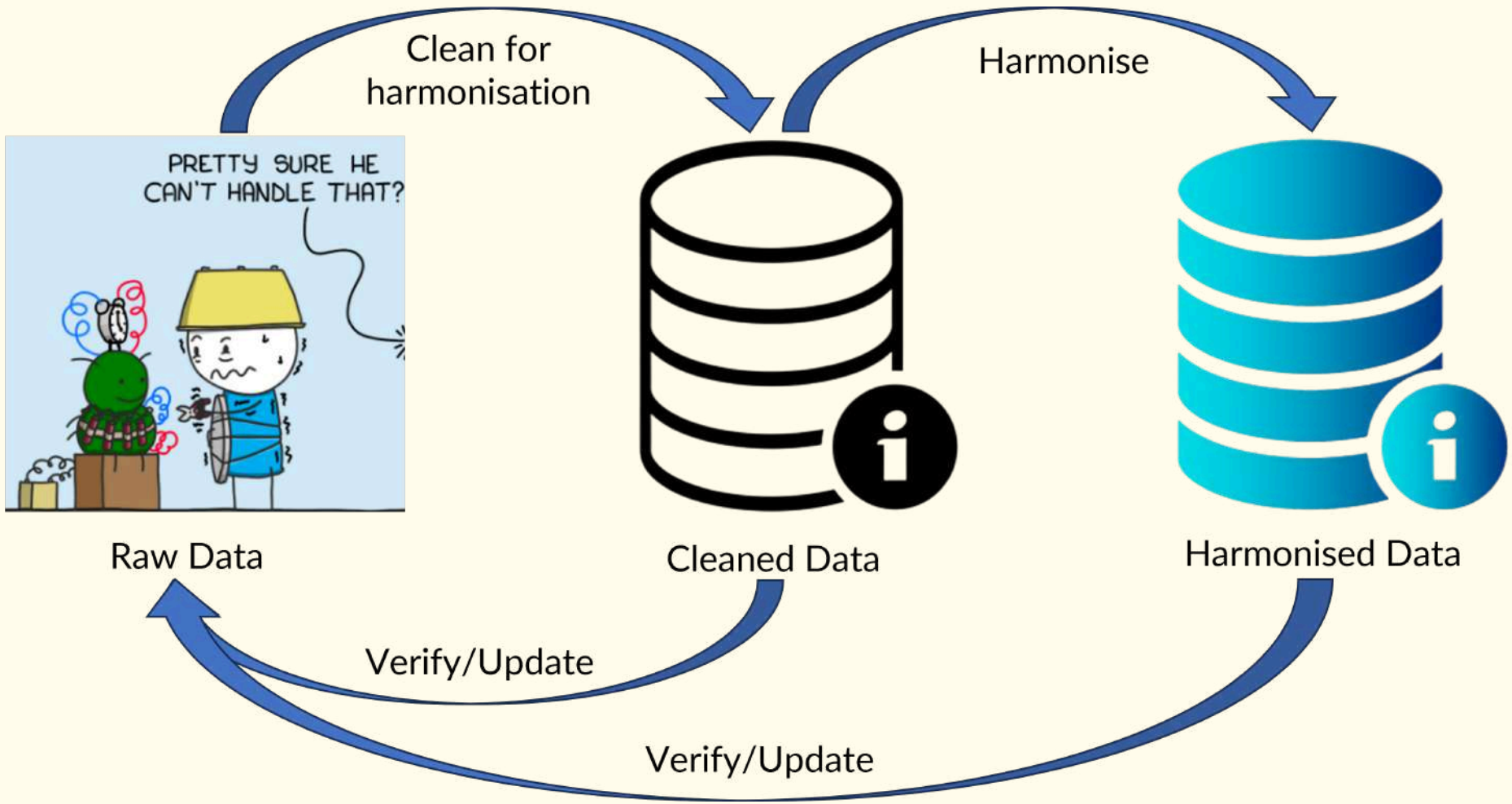
Collaborator can update the raw data. For example, adding new clinical data, add more patients, correct errors.



Dataset vectors by Vectora Artworks from Vecteezy.

Workflow with collaborators

New version means new bugs or reopen issues to fix. Is there an automated way to catch warnings/issues when reading these updated files ?



Dataset vectors by Vectora Artworks from Vecteezy. Anticipate from MonkeyUser.com

Automated capture of warnings (csv)

Is there an automated way to catch warnings/issues when reading csv files ?

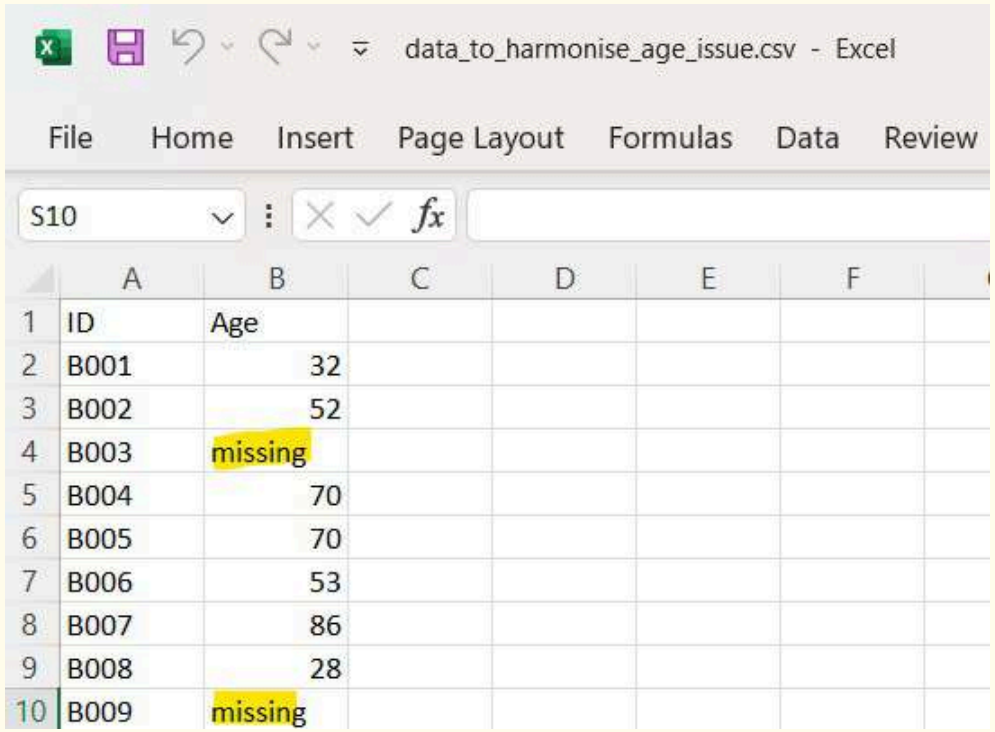
```r
cohort_data_csv <- vroom::vroom(
  file = here::here("data-raw", "Cohort_csv",
    "data_to_harmonise_age_issue.csv"),
  delim = ",",
  col_select = 1:2,
  show_col_types = FALSE,
  col_types = list(
    ID = vroom::col_character(),
    Age = vroom::col_integer()
  )
)

head(cohort_data_csv, n = 3)
```

```
# A tibble: 3 × 2

Warning: One or more parsing issues, call `problems()` on your
data frame for details,
e.g.:
  dat <- vroom(...)
  problems(dat)

  ID      Age
  <chr> <int>
1 B001     32
2 B002     52
3 B003     NA
```



Automated capture of warnings (csv)

If there are issues with the data, the output of `vroom::problems` will be a tibble.

```r
cohort_data_csv |>
  vroom::problems()
```

```
# A tibble: 4 × 5
    row   col expected   actual  file
  <int> <int> <chr>      <chr>   <chr>
1     2     2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/use…
2     4     2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/use…
3    10     2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/use…
4    17     2 an integer missing D:/Jeremy/PortableR/RPortableWorkDirectory/use…
```

To check for this automatically, we can use `pointblank::expect_row_count_match`.

```r
cohort_data_csv |>
  vroom::problems() |>
  pointblank::expect_row_count_match(count = 0)
```

```
Error: Row counts for the two tables did not match.
The `expect_row_count_match()` validation failed beyond the absolute threshold level (1).
* failure level (1) >= failure threshold (1)
```

Automated capture of warnings (csv)

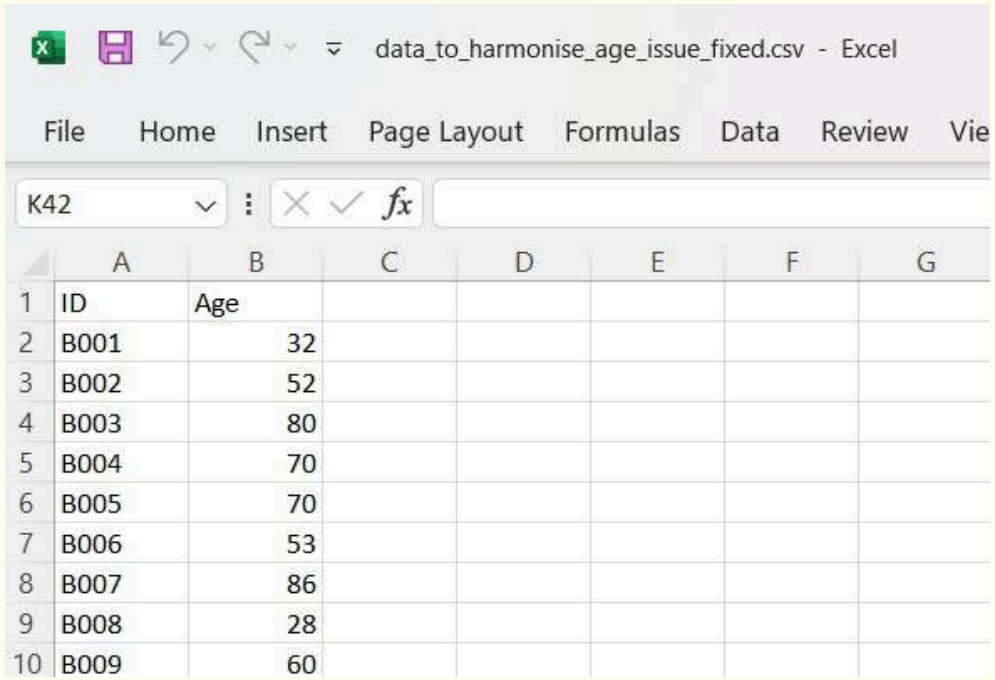
Here is a case with no issues.

```r
cohort_data_csv <- vroom::vroom(
  file = here::here("data-raw", "Cohort_csv",
    "data_to_harmonise_age_issue_fixed.csv"),
  delim = ",",
  col_select = 1:2,
  show_col_types = FALSE,
  col_types = list(
    ID = vroom::col_character(),
    Age = vroom::col_integer()
  )
)

cohort_data_csv |>
  vroom::problems()
```

```
# A tibble: 0 × 5
# ℹ 5 variables: row <int>, col <int>, expected <chr>, actual
<chr>, file <chr>
```

```r
cohort_data_csv |>
  vroom::problems() |>
  pointblank::expect_row_count_match(count = 0)
```



	A	B
1	ID	Age
2	B001	32
3	B002	52
4	B003	80
5	B004	70
6	B005	70
7	B006	53
8	B007	86
9	B008	28
10	B009	60

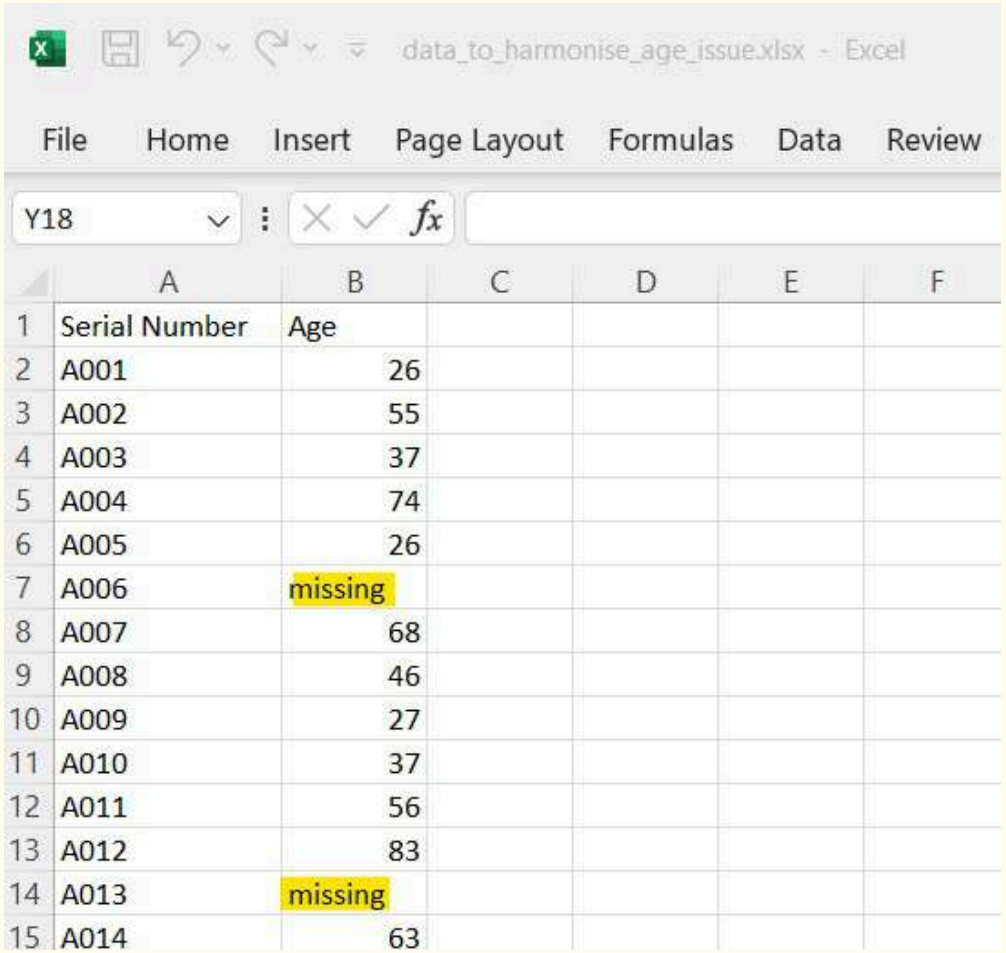
Automated capture of warnings (Excel)

Is there an automated way to catch warnings/issues when reading Excel files ?

```r
cohort_data_excel <- readxl::read_excel(
  path = here::here("data-raw", "Cohort_Excel",
    "data_to_harmonise_age_issue.xlsx"),
  sheet = "Sheet1",
  col_types = c(
    "text", "numeric"
    )
  )
```

```
Warning: Expecting numeric in B7 / R7C2: got 'missing'

Warning: Expecting numeric in B14 / R14C2: got 'missing'
```



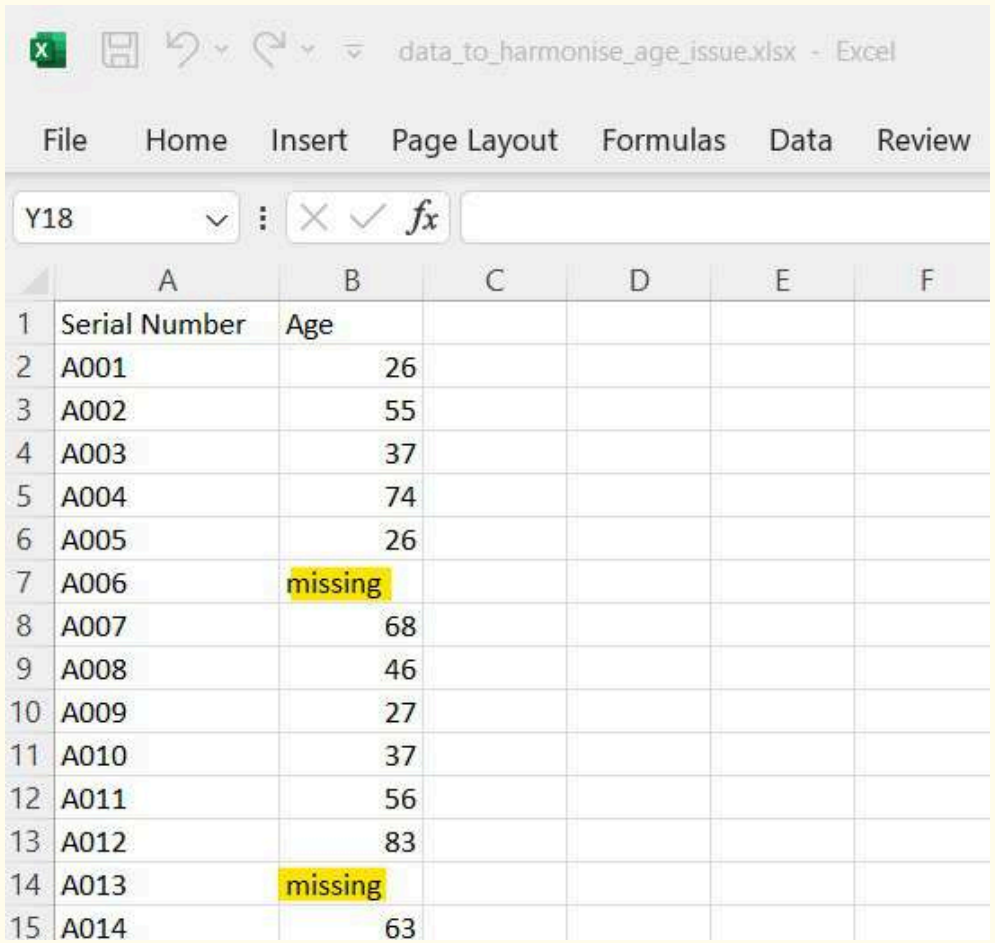
	A	B	C	D	E	F
1	Serial Number	Age				
2	A001	26				
3	A002	55				
4	A003	37				
5	A004	74				
6	A005	26				
7	A006	missing				
8	A007	68				
9	A008	46				
10	A009	27				
11	A010	37				
12	A011	56				
13	A012	83				
14	A013	missing				
15	A014	63				

Automated capture of warnings (Excel)

We can read the Excel file with `testthat::expect_no_condition`.

```r
testthat::expect_no_condition(
  cohort_data_excel <- readxl::read_excel(
    path = here::here("data-raw", "Cohort_Excel",
      "data_to_harmonise_age_issue.xlsx"),
    sheet = "Sheet1",
    col_types = c("text", "numeric")
  )
)
```

```
Error: Expected `... <- NULL` to run without any conditions.
ℹ Actually got a <simpleWarning> with text:
  Expecting numeric in B7 / R7C2: got 'missing'
```



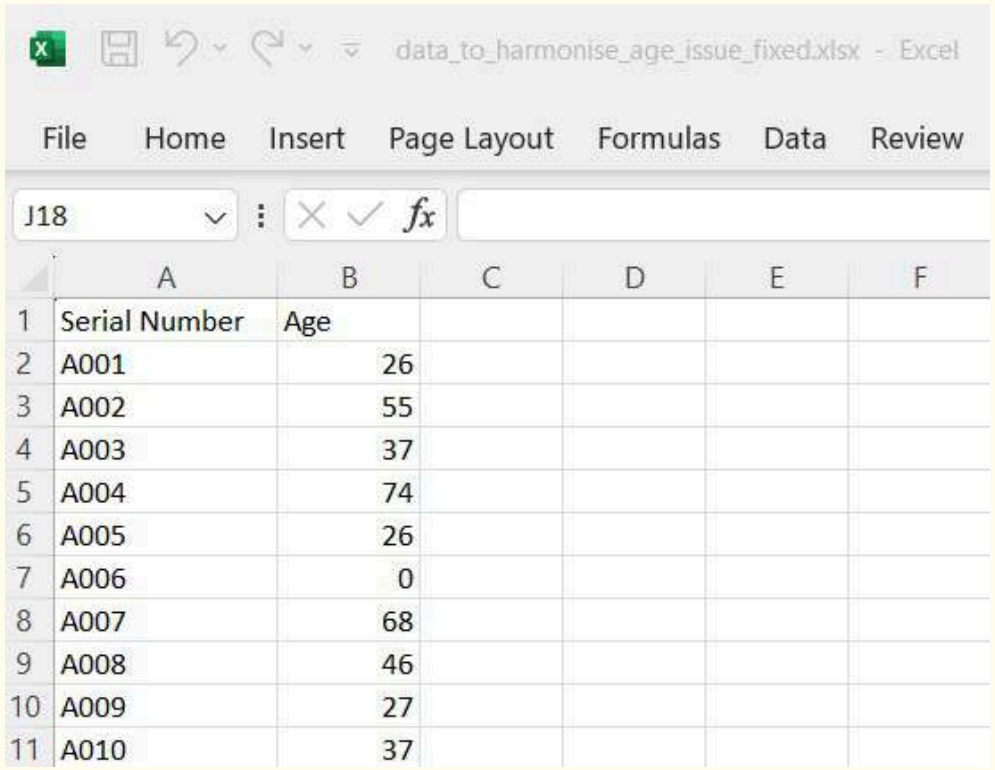
	A	B	C	D	E	F
1	Serial Number	Age				
2	A001	26				
3	A002	55				
4	A003	37				
5	A004	74				
6	A005	26				
7	A006	missing				
8	A007	68				
9	A008	46				
10	A009	27				
11	A010	37				
12	A011	56				
13	A012	83				
14	A013	missing				
15	A014	63				

Automated capture of warnings (Excel)

However, this method means that you will lose the pipe workflow.

```r
testthat::expect_no_condition(
  cohort_data_excel <- readxl::read_excel(
    path = here::here("data-raw", "Cohort_Excel",
      "data_to_harmonise_age_issue_fixed.xlsx"),
    sheet = "Sheet1",
    col_types = c("text", "numeric")
  )
)

cohort_data_excel <- cohort_data_excel |>
  # Check if Serial Number is unique
  pointblank::rows_distinct(
    columns = "Serial Number",
  )
```



	A	B	C	D	E	F
1	Serial Number	Age				
2	A001	26				
3	A002	55				
4	A003	37				
5	A004	74				
6	A005	26				
7	A006	0				
8	A007	68				
9	A008	46				
10	A009	27				
11	A010	37				

Automated capture of warnings (Excel)

We can use the tee pipe operator %T>% from magrittr.

With Issues

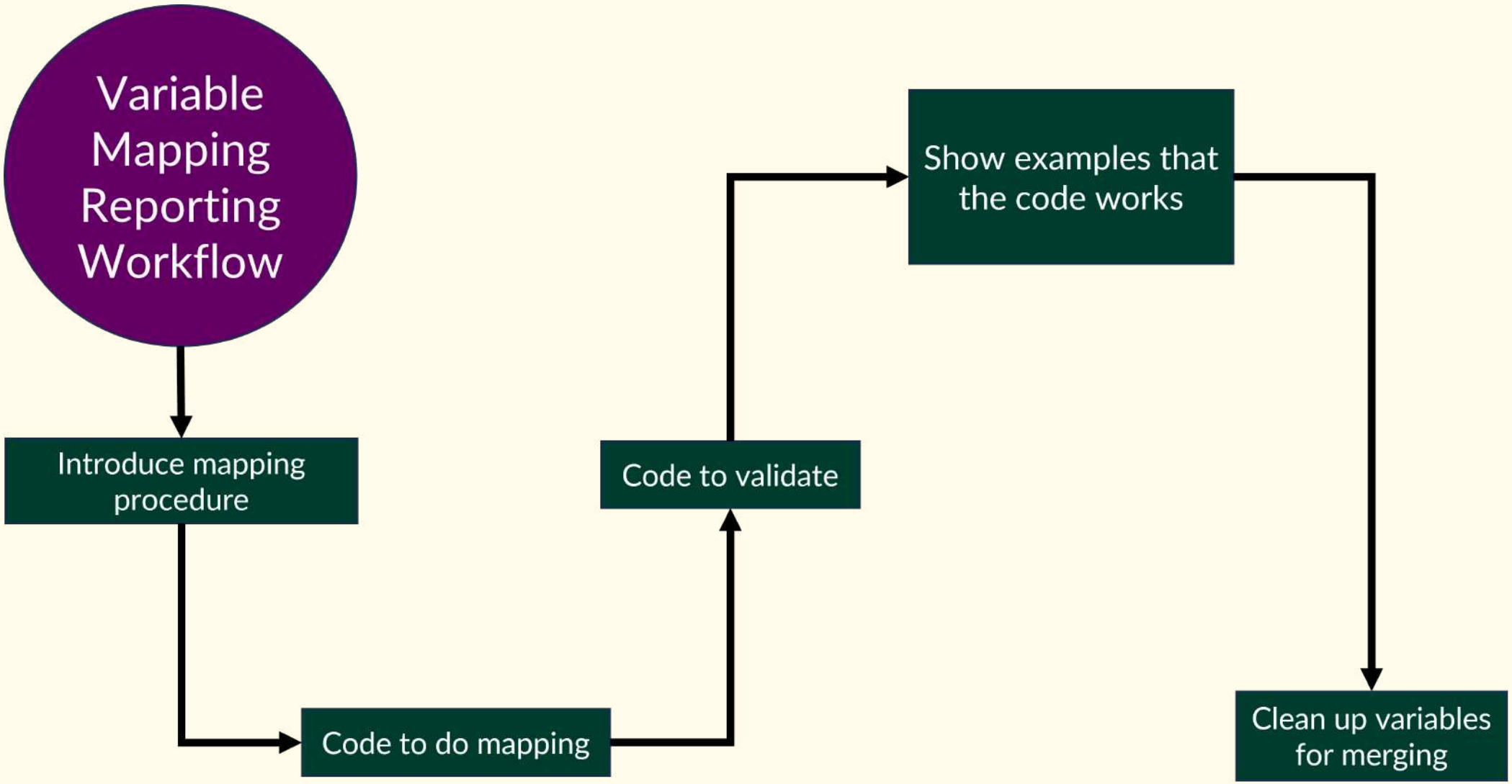
```r
cohort_data_excel <- readxl::read_excel(
  path = here::here("data-raw", "Cohort_Excel",
    "data_to_harmonise_age_issue.xlsx"),
  sheet = "Sheet1",
  col_types = c(
    "text", "numeric"
    )
  ) %T>%
  testthat::expect_no_condition()
```

```
Error: Expected `.` to run without any conditions.
ℹ Actually got a <simpleWarning> with text:
  Expecting numeric in B7 / R7C2: got 'missing'
```

No Issues

```r
cohort_data_excel_2 <- readxl::read_excel(
  path = here::here("data-raw", "Cohort_Excel",
    "data_to_harmonise_age_issue_fixed.xlsx"),
  sheet = "Sheet1",
  col_types = c("text", "numeric")
) %T>%
testthat::expect_no_condition() |>
# Check if Serial Number is unique
  pointblank::rows_distinct(
    columns = "Serial Number",
)
```

Variable Mapping



Variable Mapping

Let take this data set as an example.

```r
cohort_csv_data <- vroom::vroom(
  file = here::here("data-raw",
                     "Cohort_csv",
                     "data_to_harmonise.csv"),
  delim = ",",
  col_select = 1:8,
  show_col_types = FALSE,
  col_types = list(
    ID = vroom::col_character(),
    Age = vroom::col_integer(),
    Sex = vroom::col_character(),
    Height = vroom::col_double(),
    Weight = vroom::col_double(),
    `Smoke History` = vroom::col_character(),
    `Chest Pain Character` = vroom::col_character(),
    Dyspnea = vroom::col_character()
    )
  ) |>
  dplyr::rename(cohort_unique_id = "ID") |>
  # Remove rows when the ID value is NA
  dplyr::filter(!is.na(.data[["cohort_unique_id"]])) |>
  # Remove white spaces in column names
```

cohort_uniq ue_id	Age	Sex	Height	Weight	Sı H
		All			A
B001	32	Female	170	63	n
B002	52	Female	167	71	cu sr
B003	80	Male	184	77	n
B004	70	Male	160	83	pa
B005	70	Female	155	61	cu sr

1–5 of 20 rows Previous 1 of 4 Next

Download as CSV

Variable Mapping

Let the reader know how the collaborator's data `Smoke History` is going to be mapped.

Introduce mapping procedure

```
### Smoking History

`smoke_current` is grouped as follows:

```{r}
#| label: smoke current table
#| echo: false
#| message: false
#| warnings: false
#| results: asis

tabl <- "

+-----------------------+---------------------+
| Smoke History         | smoke_current       |
+:======================+:====================+
| non-smoker            | 0                   |
+-----------------------+---------------------+
| past smoker           | 0                   |
+-----------------------+---------------------+
| current smoker        | 1                   |
+-----------------------+---------------------+
| NA                    | -1                  |
+-----------------------+---------------------+

"
cat(tabl)
```
```

2.3 Smoking History

`smoke_current` is grouped as follows:

Smoke History	smoke_current
non-smoker	0
past smoker	0
current smoker	1
NA	-1

`smoke_past` is grouped as follows:

Smoke History	smoke_past
non-smoker	0
past smoker	1
current smoker	0
NA	-1

We do a check to ensure that we can only have these scenarios

- `smoke_current` as 1 and `smoke_past` as 0 for current smokers
- `smoke_current` as 0 and `smoke_past` as 1 for past smokers
- `smoke_current` as 0 and `smoke_past` as 0 for non-smokers
- `smoke_current` as -1 and `smoke_past` as -1 for unknown

Variable Mapping

Code to do mapping

```r
smoking_data <- cohort_csv_data |>
  dplyr::select(c("cohort_unique_id",
                  "Smoke History")) |>
  dplyr::mutate(
    smoke_current = dplyr::case_when(
      is.na(.data[["Smoke History"]]) ~ "-1",
      .data[["Smoke History"]] == "non-smoker" ~ "0",
      .data[["Smoke History"]] == "past smoker" ~ "0",
      .data[["Smoke History"]] == "current smoker" ~ "1",
      .default = NA_character_
    ),
    smoke_current = forcats::fct_relevel(
      .data[["smoke_current"]],
      c("0", "1")),
    smoke_past = dplyr::case_when(
      is.na(.data[["Smoke History"]]) ~ "-1",
      .data[["Smoke History"]] == "non-smoker" ~ "0",
      .data[["Smoke History"]] == "past smoker" ~ "1",
      .data[["Smoke History"]] == "current smoker" ~ "0",
      .default = NA_character_
    ),
    smoke_past = forcats::fct_relevel(
```

2.3 Smoking History

`smoke_current` is grouped as follows:

Smoke History	smoke_current
non-smoker	0
past smoker	0
current smoker	1
NA	-1

`smoke_past` is grouped as follows:

Smoke History	smoke_past
non-smoker	0
past smoker	1
current smoker	0
NA	-1

Variable Mapping

Code to validate

```r
smoking_data <- smoking_data |>
  pointblank::col_vals_in_set(
    columns = c("smoke_current", "smoke_past"),
    set = c("0", "1", "-1")
  ) |>
  pointblank::col_vals_expr(
    expr = pointblank::expr(
      (.data[["smoke_current"]] == "1" & .data[["smoke_past"]]
      (.data[["smoke_current"]] == "-1" & .data[["smoke_past"]]
      (.data[["smoke_current"]] == "0" & .data[["smoke_past"]]
    )
  )
```

We do a check to ensure that we can only have these scenarios

- `smoke_current` as 1 and `smoke_past` as 0 for current smokers
- `smoke_current` as 0 and `smoke_past` as 1 for past smokers
- `smoke_current` as 0 and `smoke_past` as 0 for non-smokers
- `smoke_current` as -1 and `smoke_past` as -1 for unknown

Reference: https://github.com/rstudio/pointblank/issues/578

Variable Mapping

Show examples that the code works

```
::: {.content-visible when-format="html"}

```{r}
#| label: smoking data html
#| eval: !expr out_type == "html"

if (params$show_table && knitr::is_html_output()) {
  smoking_data |>
    harmonisation::reactable_with_download_csv_button()
}

```

:::
```

Html Output

cohort_unique_id	Smoke History	smoke_current	smoke_past
	All	All	All
B001	non-smoker	0	0
B002	current smoker	1	0
B003	non-smoker	0	0
B004	past smoker	0	1
B005	current smoker	1	0

1–5 of 20 rows Previous 1 of 4 Next

Download as CSV

Variable Mapping

Show examples that the code works

```
::: {.content-visible unless-format="html"}

```{r}
#| label: smoking data not html
#| eval: !expr out_type != "html"

if (params$show_table) {
  smoking_data |>
    dplyr::distinct(.data[["Smoke History"]],
                    .keep_all = TRUE) |>
    knitr::kable()
}

```

:::
```

Pdf Output

```
if (params$show_table) {
  smoking_data |>
    dplyr::distinct(.data[["Smoke History"]],
                    .keep_all = TRUE) |>
    knitr::kable()
}
```

cohort_unique_id	Smoke History	smoke_current	smoke_past
B001	non-smoker	0	0
B002	current smoker	1	0
B004	past smoker	0	1
B017	NA	-1	-1

Variable Mapping

Clean up variables for merging

```r
smoking_data <- smoking_data |>
  dplyr::select(-c("Smoke History"))
```

cohort_unique_id	smoke_current	smoke_past
	All	All
B001	0	0
B002	1	0
B003	0	0
B004	0	1
B005	1	0

1–5 of 20 rows Previous 1 of 4 Next

Download as CSV

Merging Harmonised Data

Suppose we have completed harmonising a batch of clinical data.

```r
age_gender_data |>
  reactable_with_download_csv_button(
    defaultPageSize = 5,
    paginationType = "jump",
    style = list(fontSize = "1rem"),
  )
```

cohort_unique_id	age_years	sex
		All
B001	32	0
B002	52	0
B003	80	1
B004	70	1
B005	70	0

1–5 of 20 rows Previous 1 of 4 Next

Download as CSV

```r
body_measurement_data |>
  reactable_with_download_csv_button(
    defaultPageSize = 5,
    paginationType = "jump",
    style = list(fontSize = "1rem"),
  )
```

cohort_unique_id	height_cm	weight_kg	bsa_m2	bmi
B001	170	63	1.72	21.8
B002	167	71	1.81	25.46
B003	184	77	1.98	22.74
B004	160	83	1.92	32.42
B005	155	61	1.62	25.39

1–5 of 20 rows Previous 1 of 4 Next

Download as CSV

How can we merge them without issues of missing rows or additional columns ?

Merging Harmonised Data

`unmatched = "error"` in `dplyr::inner_join` helps to avoid patients with no match.

```r
join_specification <- dplyr::join_by("cohort_unique_id")

demo_behave_data <- cohort_csv_data |>
  dplyr::select(c("cohort_unique_id")) |>
  dplyr::inner_join(age_gender_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::inner_join(body_measurement_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::inner_join(smoking_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::relocate(c("bsa_m2", "bmi"),
                  .after = "sex")
```

```r
three_penguins <- tibble::tribble(
  ~samp_id, ~species,    ~island,
  1,        "Adelie",    "Torgersen",
  2,        "Gentoo",    "Biscoe",
)

weight_extra <- tibble::tribble(
  ~samp_id,  ~body_mass_g,
  1,         3220,
  2,         4730,
  4,         4725
)

three_penguins |>
  dplyr::inner_join(
    y = weight_extra,
    by = dplyr::join_by("samp_id"),
    unmatched = "error"
 )
```

```
Error in `dplyr::inner_join()`:
! Each row of `y` must be matched by `x`.
i Row 3 of `y` was not matched.
```

Reference: https://www.tidyverse.org/blog/2023/08/teach-tidyverse-23/#improved-and-expanded-_join-functionality

Merging Harmonised Data

`unmatched = "error"` in `dplyr::inner_join` helps to avoid patients with no match.

```r
join_specification <- dplyr::join_by("cohort_unique_id")

demo_behave_data <- cohort_csv_data |>
  dplyr::select(c("cohort_unique_id")) |>
  dplyr::inner_join(age_gender_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::inner_join(body_measurement_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::inner_join(smoking_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::relocate(c("bsa_m2", "bmi"),
                  .after = "sex")
```

```r
three_penguins <- tibble::tribble(
  ~samp_id, ~species,    ~island,
  1,        "Adelie",    "Torgersen",
  2,        "Gentoo",    "Biscoe",
  3,        "Chinstrap", "Dream"
)

weight_extra <- tibble::tribble(
  ~samp_id,  ~body_mass_g,
  1,         3220,
  3,         4725
)

three_penguins |>
  dplyr::inner_join(
    y = weight_extra,
    by = dplyr::join_by("samp_id"),
    unmatched = "error"
 )
```

```
Error in `dplyr::inner_join()`:
! Each row of `x` must have a match in `y`.
ℹ Row 2 of `x` does not have a match.
```

Reference: https://www.tidyverse.org/blog/2023/08/teach-tidyverse-23/#improved-and-expanded-_join-functionality

Merging Harmonised Data

`relationship = "one-to-one"` in `dplyr::inner_join` helps to avoid patients with multiple match.

```r
join_specification <- dplyr::join_by("cohort_unique_id")

demo_behave_data <- cohort_csv_data |>
  dplyr::select(c("cohort_unique_id")) |>
  dplyr::inner_join(age_gender_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::inner_join(body_measurement_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::inner_join(smoking_data,
                    by = join_specification,
                    unmatched = "error",
                    relationship = "one-to-one") |>
  dplyr::relocate(c("bsa_m2", "bmi"),
                  .after = "sex")
```

```r
three_penguins <- tibble::tribble(
  ~samp_id, ~species,    ~island,
  1,        "Adelie",    "Torgersen",
  2,        "Gentoo",    "Biscoe",
  3,        "Chinstrap", "Dream"
)

weight_extra <- tibble::tribble(
  ~samp_id,  ~body_mass_g,
  1,         3220,
  2,         4730,
  2,         4725,
  3,         4000
)

three_penguins |>
  dplyr::inner_join(
    y = weight_extra,
    by = dplyr::join_by("samp_id"),
    relationship = "one-to-one"
 )
```

```
Error in `dplyr::inner_join()`:
! Each row in `x` must match at most 1 row in `y`.
ℹ Row 2 of `x` matches multiple rows in `y`.
```

Reference: https://www.tidyverse.org/blog/2023/08/teach-tidyverse-23/#improved-and-expanded-_join-functionality

Merging Harmonised Data

Use `pointblank::has_columns` to ensure we only have harmonised variables.

```r
testthat::expect_false(
  pointblank::has_columns(
    demo_behave_data,
    columns = c(
      dplyr::ends_with(".x"),
      dplyr::ends_with(".y")
    )
  )
)

testthat::expect_equal(
  ncol(demo_behave_data), 9
)

testthat::expect_true(
  pointblank::has_columns(
    demo_behave_data,
    columns = c(
      "age_years", "sex",
      "height_cm", "weight_kg", "bsa_m2", "bmi",
      "smoke_current", "smoke_past"
    )
```

```r
three_penguins <- tibble::tribble(
  ~samp_id, ~species,    ~island,
  1,        "Adelie",    "Torgersen",
  2,        "Gentoo",    "Biscoe",
  3,        "Chinstrap", "Dream"
)

weight_extra <- tibble::tribble(
  ~samp_id,  ~island,
  1,         "Torgersen",
  2,         "Biscoe",
  3,         "Dream"
)

three_penguins <- three_penguins |>
  dplyr::inner_join(
    y = weight_extra,
    by = dplyr::join_by("samp_id"),
    unmatched = "error",
    relationship = "one-to-one"
 )
```

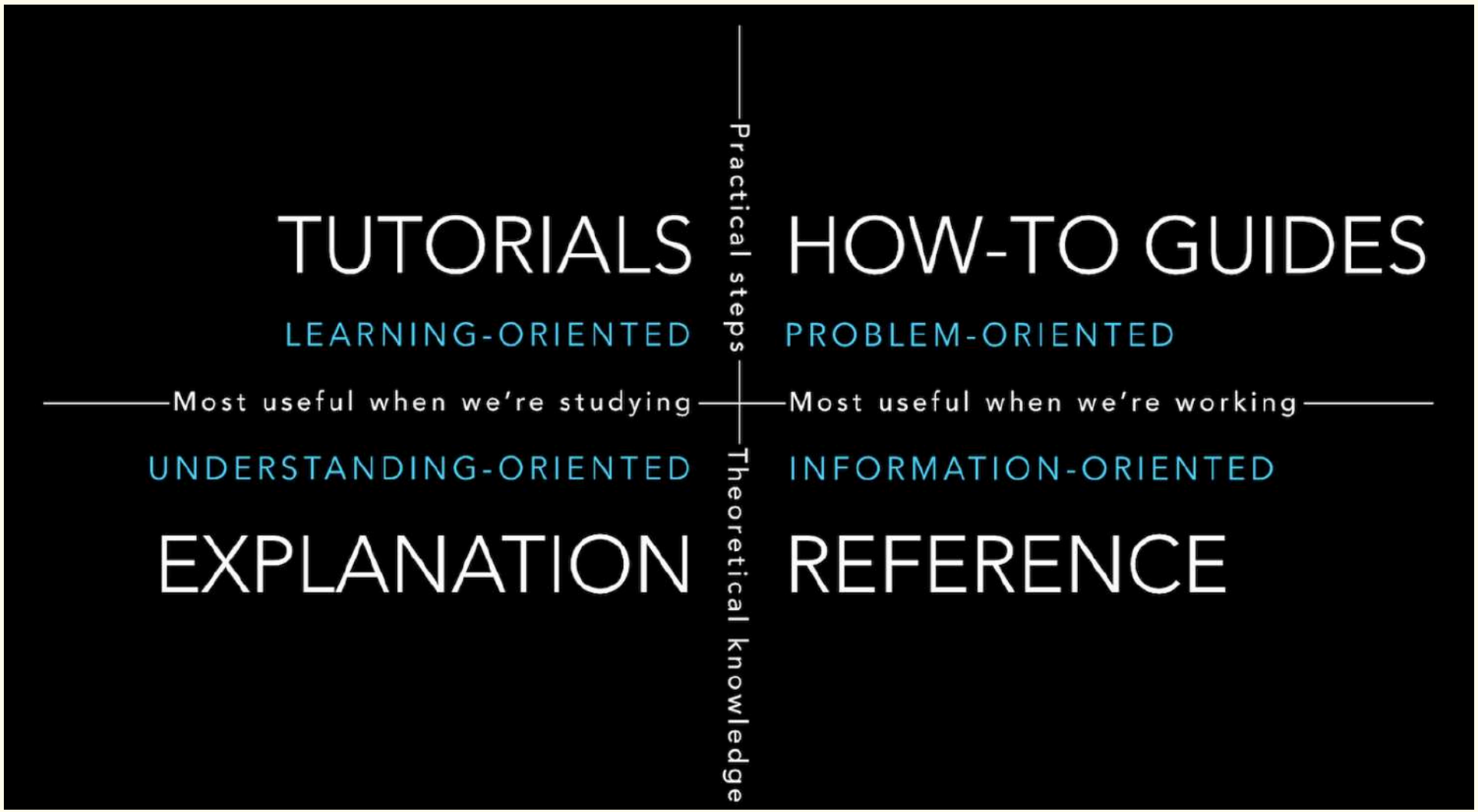
```
[1] TRUE
```

```r
colnames(three_penguins)
```

```
[1] "samp_id"  "species"  "island.x" "island.y"
```

Harmonisation Report Types

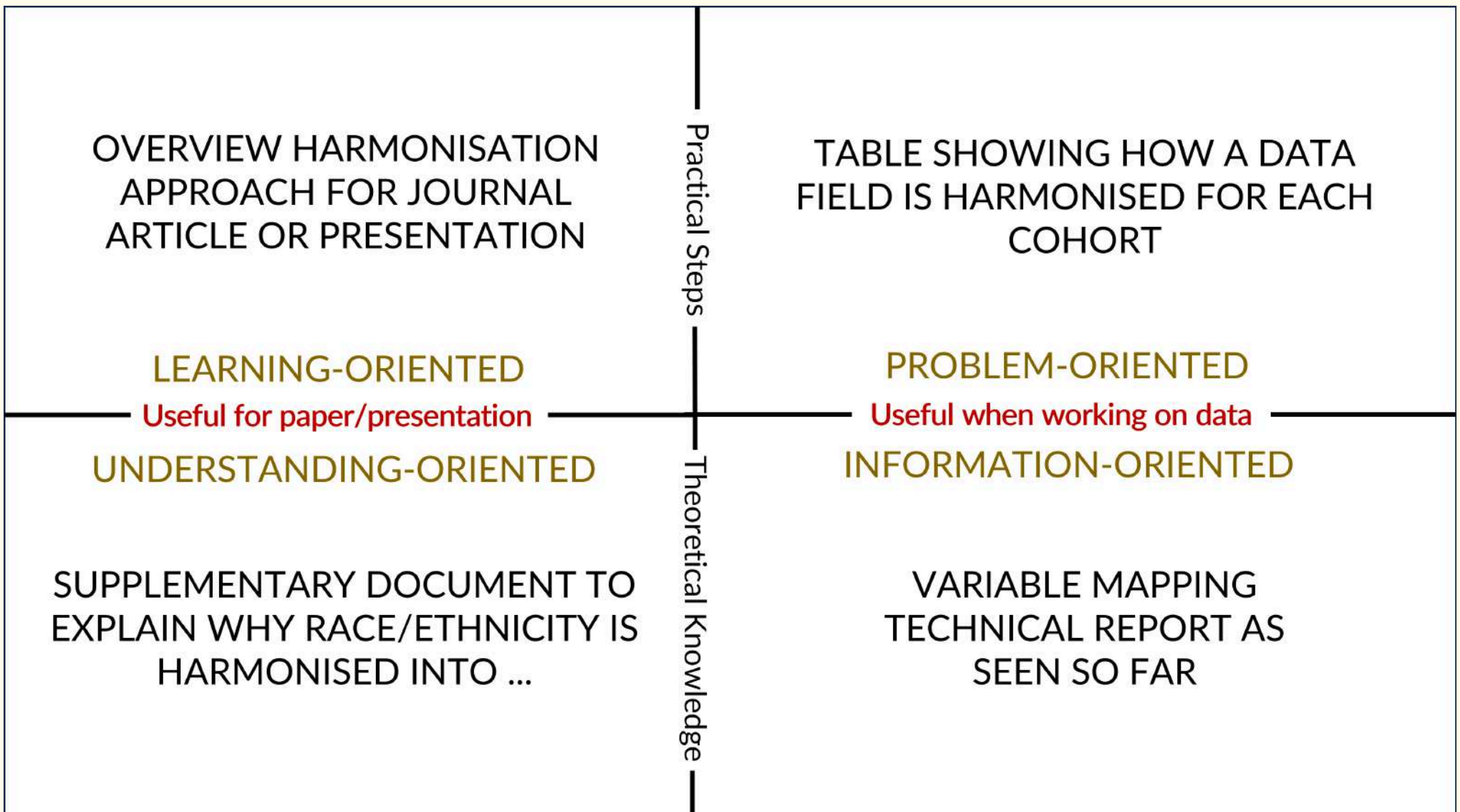
Collaborator wants different ways to report how data harmonisation is done.



The documentation system by Divio

Harmonisation Report Types

Collaborator wants different ways to report how data harmonisation is done.



Tehcncal Report Challenge

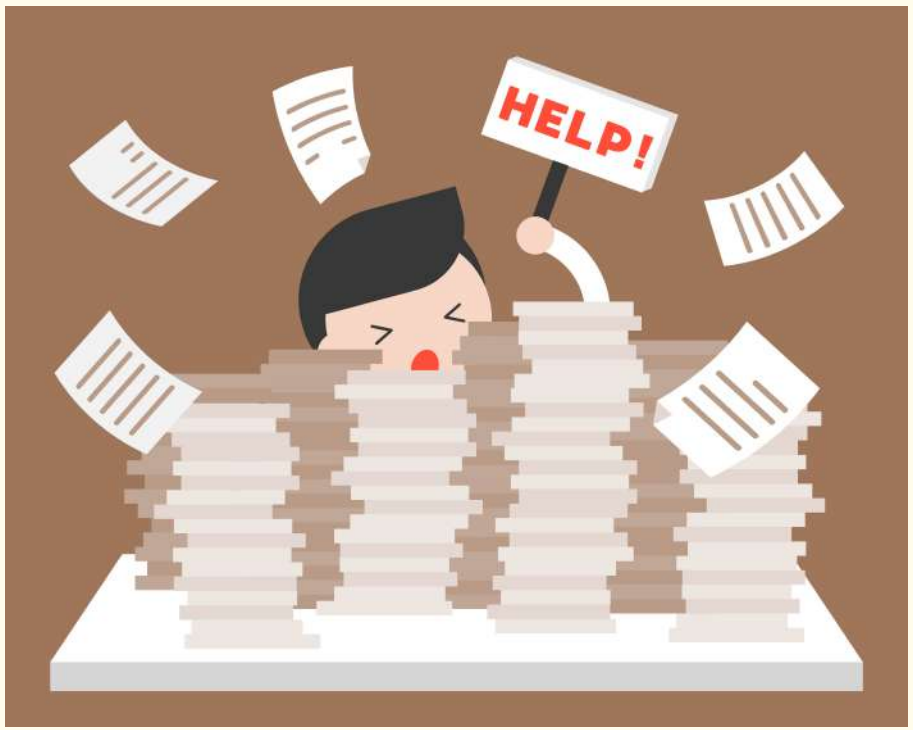
One variable mapping report takes at least one page.

On average, a clinical trial will have a few hundred variables.

- One hundred columns for clinical and demographics.
- Two hundred columns for medication.

Harmonisation report can have at least a few hundreds pages for each cohort.

There is a need to automate the creation of these reports.



Businessman in pile of documents asking for help by Amonrat Rungreangfangsai

Quarto Books

To make a Quarto book or website, we need a `_quarto.yml` and `index.qmd` file

_quarto.yml × index.qmd ×

```yaml
project:
  type: book
  output-dir: reports/Cohort_B

book:
  downloads: [pdf, docx]
  title: "Harmonisation Template for Cohort B"
  author: "My Name"
  navbar:
    search: true
  sidebar:
    collapse-level: 1

  chapters:
    - index.qmd
    - part: Cohort B Cleaning
      chapters:
        - codes/Cohort_B/00_R_Package_And_Environment.qmd
        - codes/Cohort_B/01_Read_Cohort_B_Data.qmd
        - codes/Cohort_B/02_Extract_Demographic.qmd
        - codes/Cohort_B/03_Export_To_Excel.qmd
```

_quarto.yml × index.qmd ×

Render on Save | Render | Run | Publis

Source | Visual | Outli

```yaml
---
date: "2025-03-10"
format:
  html:
    code-fold: true
    freeze: false
params:
  show_table: TRUE
---

```{r}
#| label: output type
#| echo: false
#| warning: false
#| message: false

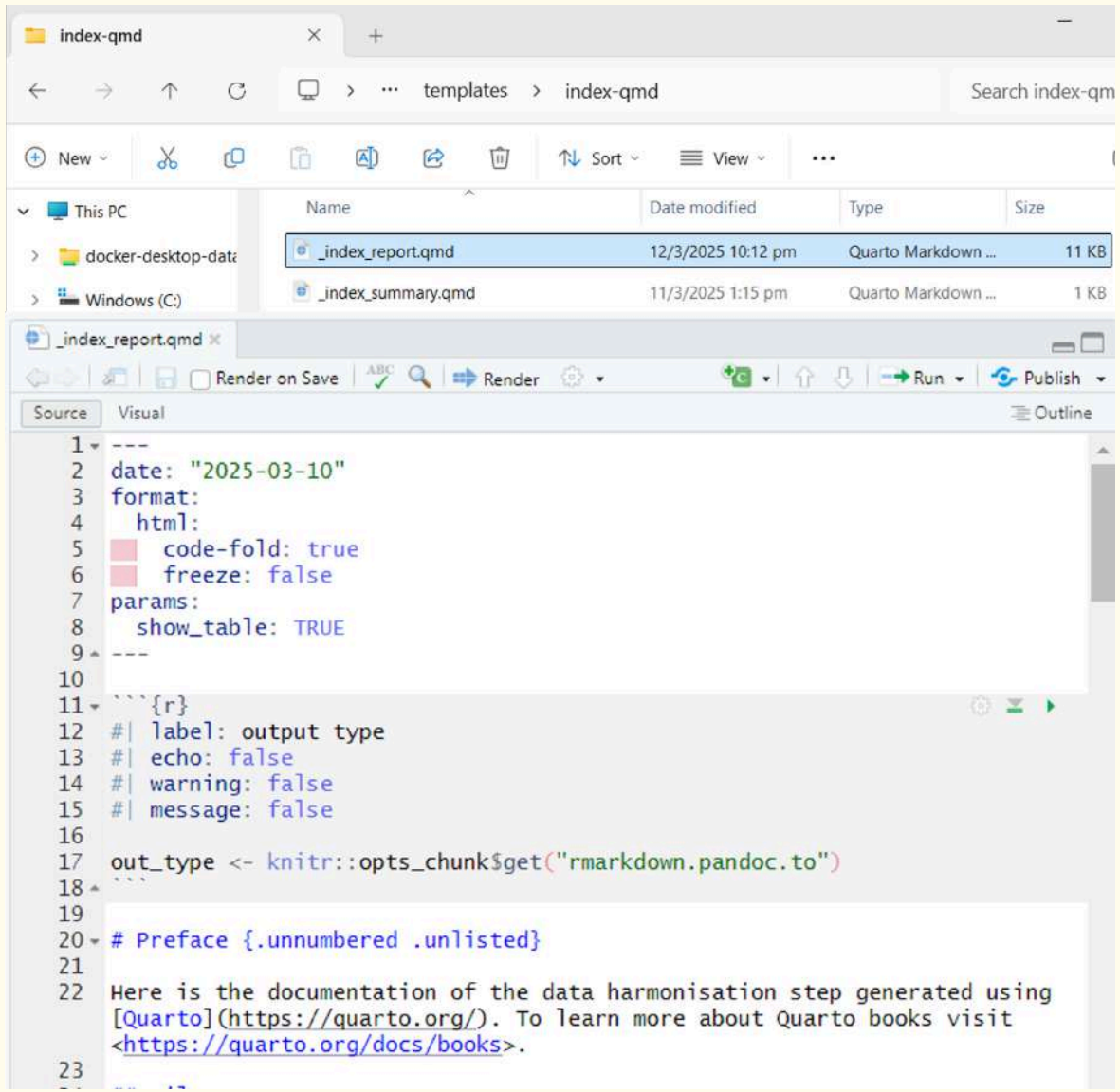
out_type <- knitr::opts_chunk$get("rmarkdown.pandoc.to")
```

Preface {.unnumbered .unlisted}

Here is the documentation of the data harmonisation step generated using [Quarto](https://quarto.org/). To learn more about Quarto books visit <https://quarto.org/docs/books>.
```

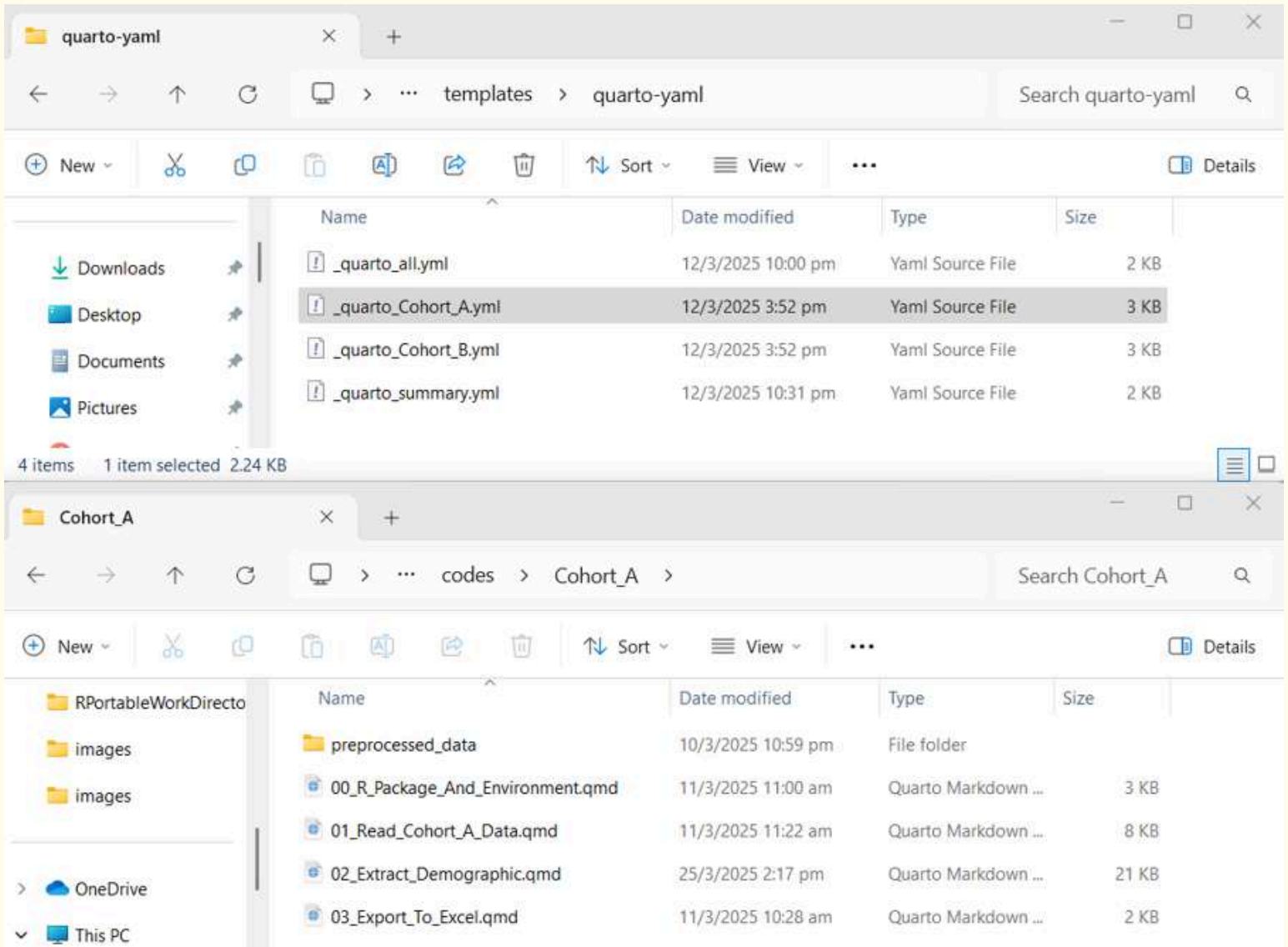
Automated Technical Report (Reference)

We create an `index.qmd` file for technical report generation.



Automated Technical Report (Reference)

We create a `_quarto.yml` file and relevant Quarto files for each cohort.



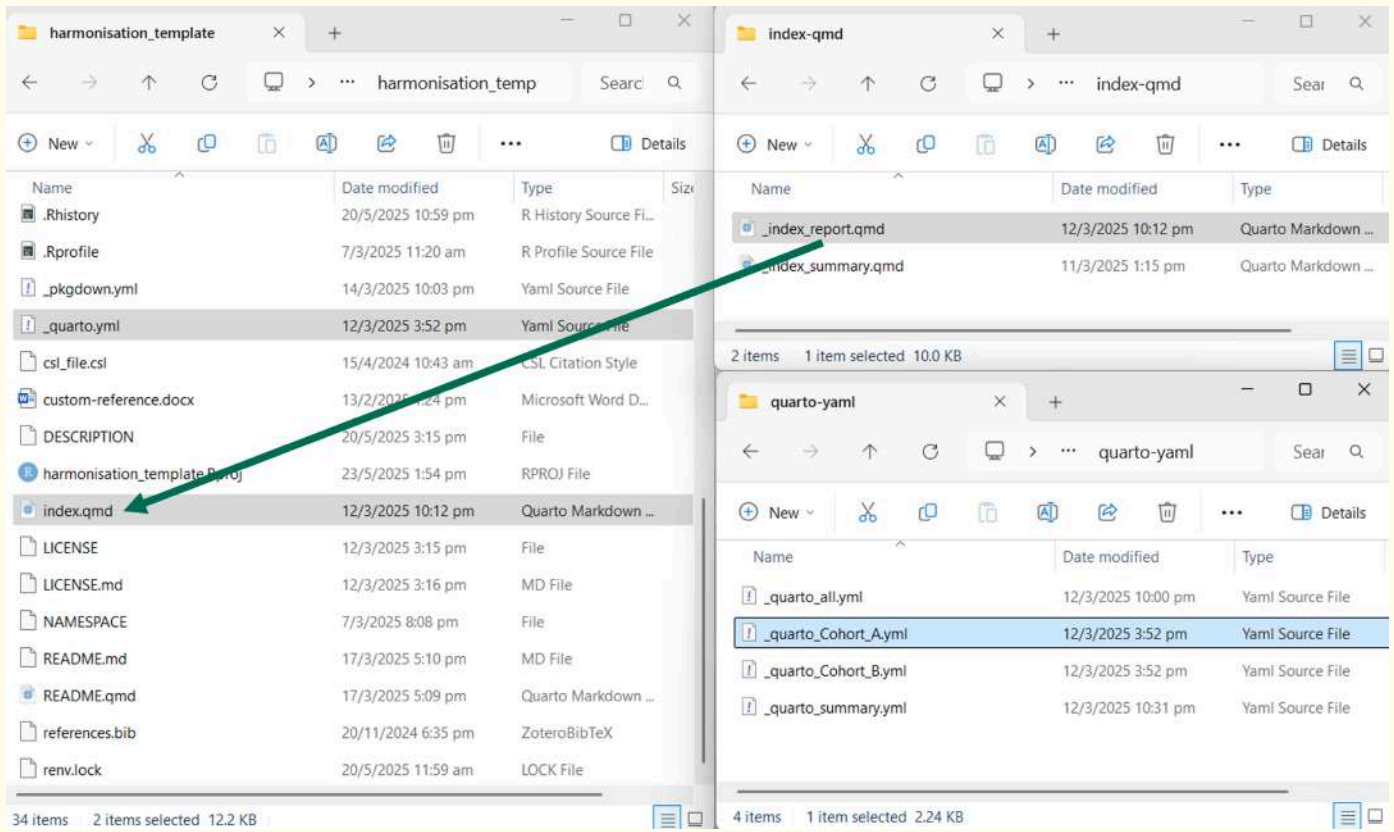
Automated Technical Report (Reference)

Create a script to generate technical reports in pdf, word and html for each cohort.

```r
# Copy the right index.qmd
# file

index_qmd_file <- paste0(
  "_index_",
  "report",
  ".qmd"
)

fs::file_copy(
  path = here::here(
    "templates",
    "index-qmd",
    index_qmd_file),
  new_path = here::here(
    "index.qmd"
  ),
  overwrite = TRUE
)
```



Automated Technical Report (Reference)

Create a script to generate technical reports in pdf, word and html for each cohort.

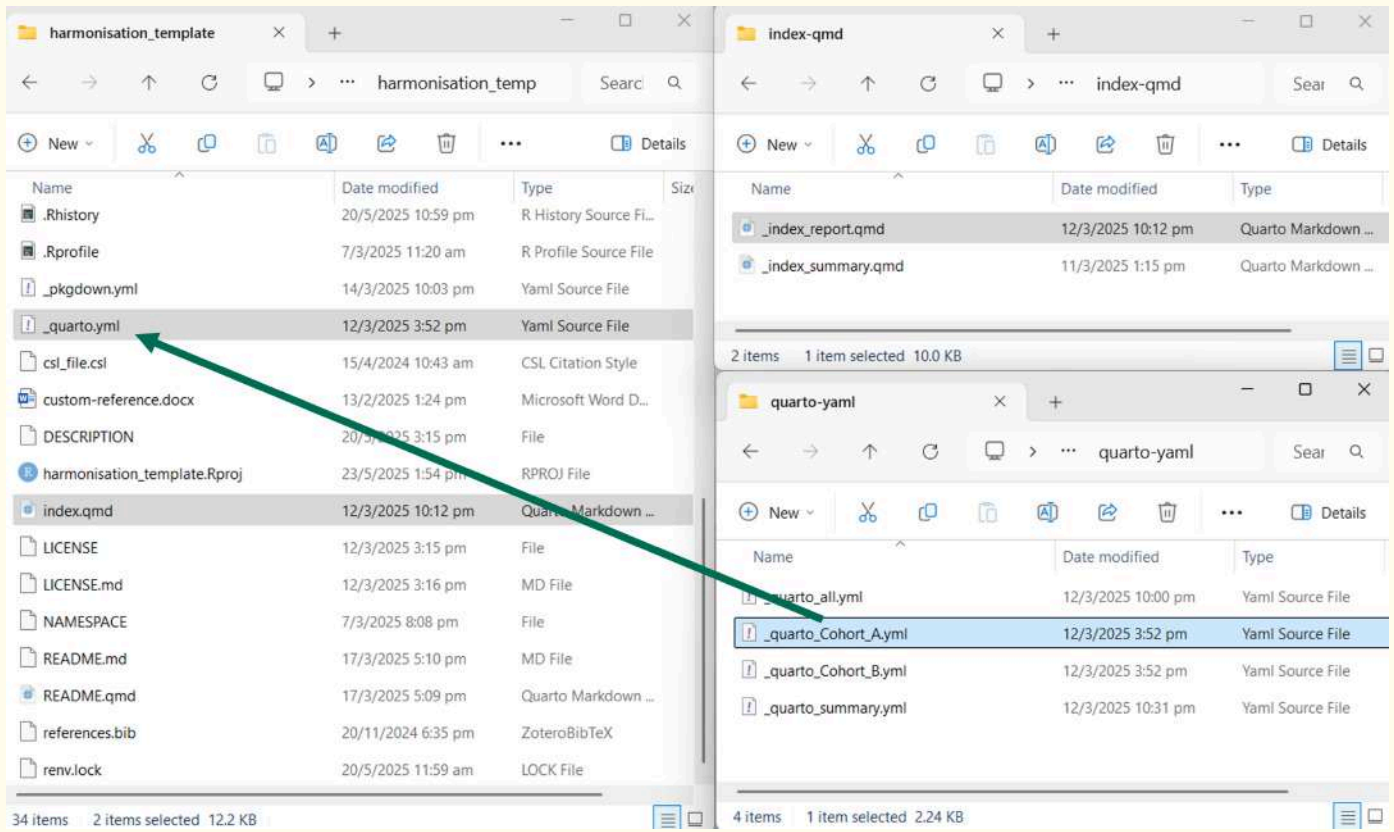
```r
copy_and_render <- function(
    cohort
) {

  # Copy quarto.yml file
  # for each cohort

  quarto_yml_file <- paste0(
    "_quarto_",
    cohort,
    ".yml"
  )

  fs::file_copy(
    path = here::here(
      "templates",
      "quarto-yaml",
      quarto_yml_file),
    new_path = here::here("_quarto
    overwrite = TRUE
  )

```

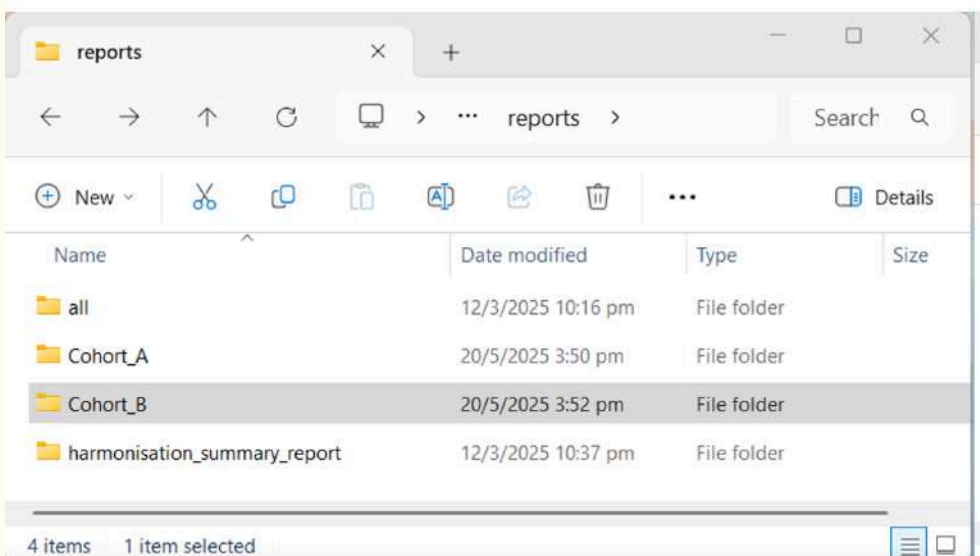


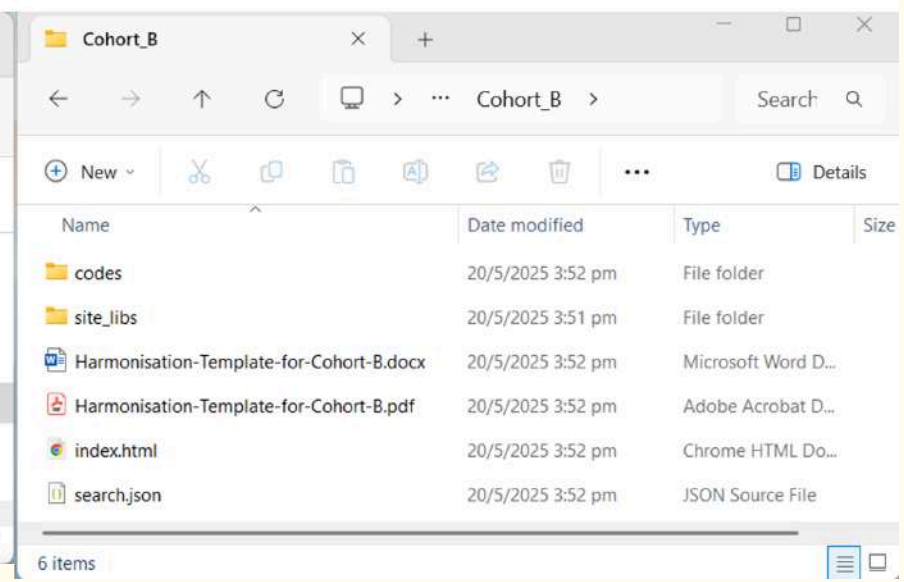
Automated Technical Report (Reference)

Output of these reports are as follows:

Run the R script `cohort_harmonisation_script.R` in `codes` folder to generate:

- Cohort_A Harmonisation Report:
  - 🌐 HTML: https://jauntyjjs-harmonisation-cohort-a.netlify.app
  - 📕 PDF : https://jauntyjjs-harmonisation-cohort-a.netlify.app/Harmonisation-Template-for-Cohort-A.pdf
  - 📘 Word: https://jauntyjjs-harmonisation-cohort-a.netlify.app/Harmonisation-Template-for-Cohort-A.docx
- Cohort_B Harmonisation Report:
  - 🌐 HTML: https://jauntyjjs-harmonisation-cohort-b.netlify.app
  - 📕 PDF : https://jauntyjjs-harmonisation-cohort-b.netlify.app/Harmonisation-Template-for-Cohort-B.pdf
  - 📘 Word: https://jauntyjjs-harmonisation-cohort-b.netlify.app/Harmonisation-Template-for-Cohort-B.docx





Automated Summary Report (How-to-Guide)

A similar method is done to create a summary report in word using `flextable`.

2.4 Smoking History

smoke_current is the harmonised data field to denote if the patient is a current smoker during the time of the CT scan. *smoke_past* is the harmonised data field to denote if the patient is a past smoker during the time of the CT scan.

They hold the following values:

Table S6: Harmonised values of smoke_current and smoke_past.

Value	Description
0	no
1	yes
-1	unknown

They are harmonised as follows:

Table S7: Harmonised process of smoke_current and smoke_past.

Cohort ID	Original Response	Harmonisation Response
Cohort A	Column *smoke_current_good* with *0* as no. *1* as yes. *-1* as unknown. Column *smoke_past_good* with *0* as no. *1* as yes. *-1* as unknown.	*smoke_current* will take the values of *smoke_current_good*. *smoke_past* will take the values of *smoke_past_good*.
Cohort B	Column *Smoke History* with *non-smoker* as non-smoker. *past smoker* as a past smoker. *current smoker* as a current smoker. *NA* as unknown.	Map the values of *Smoke History* to *smoke_current* as follows: *non-smoker* and *past smoker* as 0. *current smoker* as 1. *NA* as -1. Map the values of *Smoke History* to *smoke_past* as follows: *non-smoker* and *current smoker* as 0. *past smoker* as 1. *NA* as -1.

After harmonisation, we validate the values of *smoke_current* and *smoke_past* to ensure that there can only be the following cases:

Table S8: Valid values of smoke_current and smoke_past.

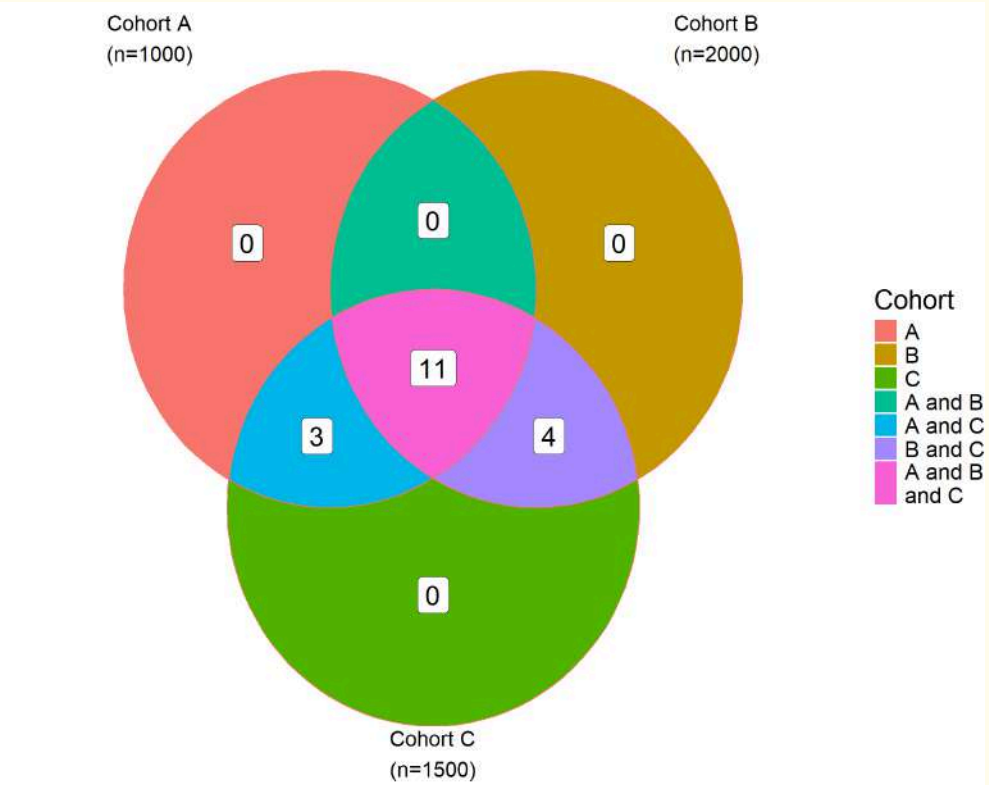
Description	*smoke_current*	*smoke_past*
Non-smoker	0	0
Past smoker	0	1
Current smoker	1	0
Unknown	-1	-1

Overview Diagrams

How many variables can each cohort provide ?

How many variables can be harmonised ?

```r
demographic_list <- list(
  A = c("Age", "Sex",
        "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabe
        "Smoke Current", "Smoke Past",
        "Have Chest Pain", "Chest Pain Character",
        "Dyspnea",
        "BMI", "Height", "Weight"),
  B = c("Age", "Sex",
        "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabe
        "Smoke Current", "Smoke Past",
        "Have Chest Pain", "Chest Pain Character",
        "Dyspnea",
        "HDL", "Total Cholesterol",
        "Triglyceride", "LDL"),
  C = c("Age", "Sex",
        "Hypertension", "Dyslipidemia", "Family Hx CAD", "Diabe
        "Smoke Current", "Smoke Past",
        "Have Chest Pain", "Chest Pain Character",
        "Dyspnea",
        "BMI", "Height", "Weight",
        "HDL", "Total Cholesterol",
        "Triglyceride", "LDL")
```

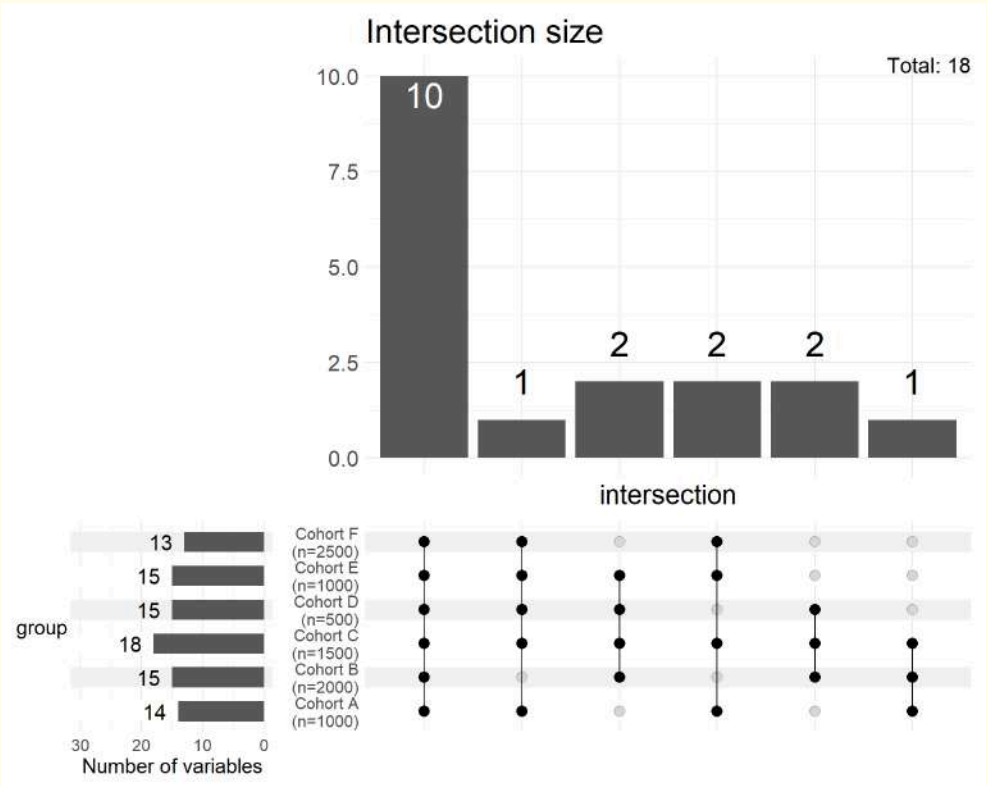


Venn diagram does not work for many (> 10) cohorts.

Overview Diagrams

Upset plots are too complicated for clinicians.

```r
demographic_venn <- tibble::tibble(
  column_name = c("Age", "Sex",
                  "Hypertension", "Dyslipidemia", "Family Hx CA
                  "Smoke Current", "Smoke Past",
                  "Have Chest Pain", "Chest Pain Character",
                  "Dyspnea",
                  "BMI", "Height", "Weight",
                  "HDL", "Total Cholesterol",
                  "Triglyceride", "LDL"),
  `Cohort A` = c(1, 1,
                 1, 1, 1, 1,
                 1, 1,
                 1, 1,
                 1,
                 1, 1, 1,
                 0, 0,
                 0, 0),
  `Cohort B` = c(1, 1,
                 1, 1, 1, 1,
                 1, 1,
                 1, 1,
                 1,
```



Cannot answer follow-up questions:

How many cohorts provide patient's blood lipid information and how many patients have them ?

Overview Diagrams

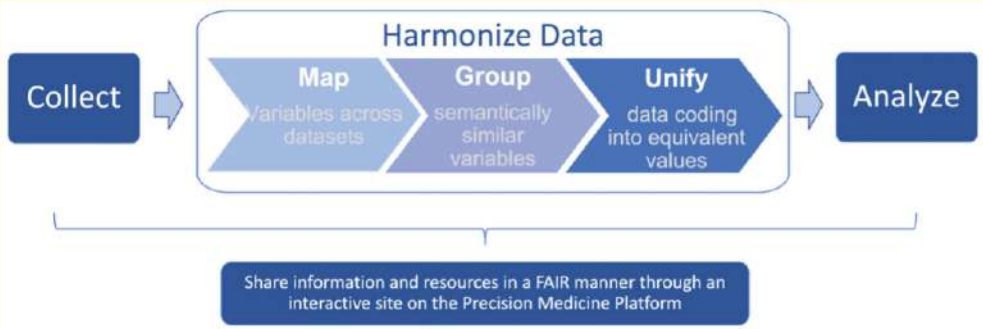
Create a “heatmap”using Microsoft PowerPoint.

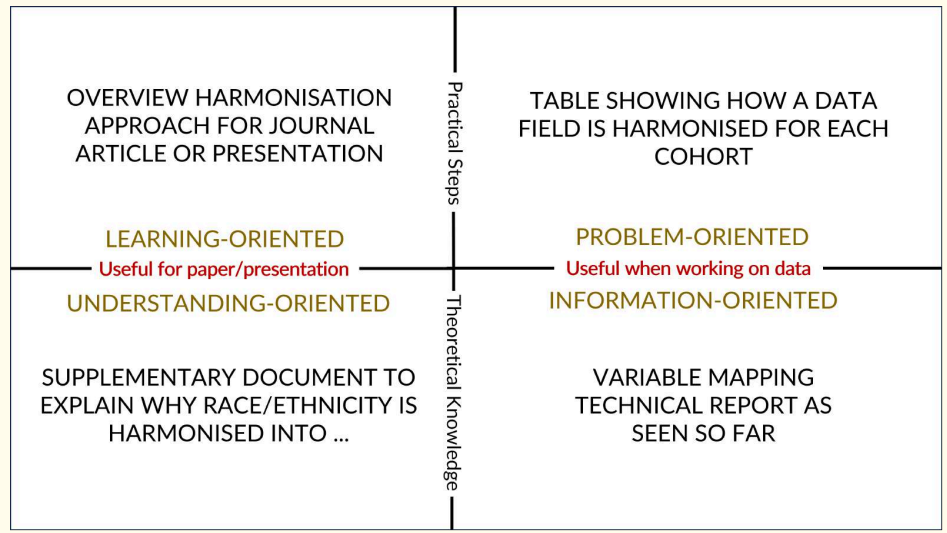
Country	Cohort	N	Variables	Age, Sex, Hypertension, Dyslipidemia, Family Hx CAD, Diabetes, Smoke Current, Smoke Past, Have Chest Pain, Chest Pain Character (10)	Dyspnea (1)	BMI (1)	Height, Weight (2)	HDL, Total Cholesterol (2)	Triglyceride, LDL (2)
Country A	Cohort A	1000	15	✓	✓	✓	✓	×	×
	Cohort B	2000	16	✓	✓	×	×	✓	✓
Country B	Cohort C	1500	18	✓	✓	✓	✓	✓	✓
Country C	Cohort D	500	16	✓	×	✓	×	✓	✓
Country D	Cohort E	1000	16	✓	×	✓	✓	✓	×
	Cohort F	2500	14	✓	×	✓	✓		
Total				**8500**	**4500**	**6500**	**6000**	**5000**	**4000**

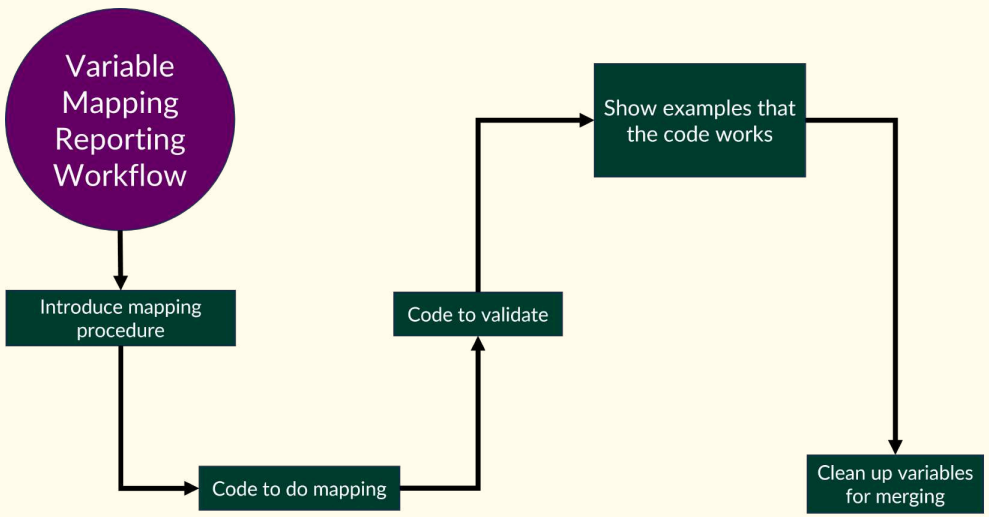
Variable Colour Legend	
	Age
	Sex
	Comorbidity
	Smoking history
	Symptoms
	Obesity
	Blood lipid

Table Legend	
✓	Available
×	Not available
	Pending arrival

Summary







				10	1	1	2	2	2
				Age					
				Sex					
				Hypertension					
				Dyslipidemia					
				Family Hx CAD					
				Diabetes					
				Smoke Current					
				Smoke Past					
				Have Chest Pain			Height	HDL	Triglyceride
Country	**Cohort**	**N**	**Variables**	Chest Pain Character	Dyspnea	BMI	Weight	Total Cholesterol	LDL
Country A	Cohort A	1000	15	✓	✓	✓	✓	×	×
	Cohort B	2000	16	✓	✓	×	×	✓	✓
Country B	Cohort C	1500	18	✓	✓	✓	✓	✓	✓
Country C	Cohort D	500	16	✓	×	✓	×	✓	✓
Country D	Cohort E	1000	16	✓	×	✓	✓	✓	×
	Cohort F	2500	14	✓	×	✓	✓		
Total				8500	4500	6500	6000	5000	4000

Variable Colour Legend	
	Age
	Sex
	Comorbidity
	Smoking history
	Symptoms
	Obesity
	Blood lipid

Table Legend	
✓	Available
×	Not available
	Pending arrival

Thank you

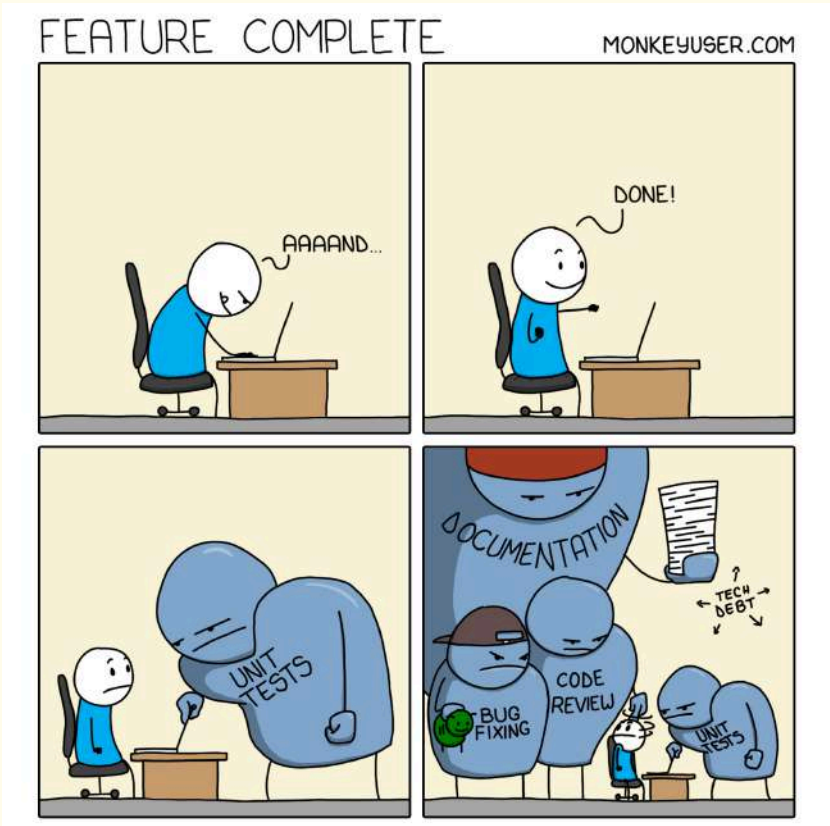
Harmonisation project template: https://github.com/JauntyJJS/harmonisation/

README License MIT license

Data Harmonisation Project Template

R-CMD-check.yaml passing

Table of Content



Feature Complete from MonkeyUser.com

Happy 60th Birthday Singapore

Singapore National Day Parade (NDP) 2025 Logo from https://www.ndp.gov.sg